Systematische Modellierung und Analyse verteilter Automatisierungssysteme

Von der Fakultät für Maschinenbau
der Technischen Universität Carolo-Wilhelmina zu Braunschweig

zur Erlangung der Würde

eines Doktor-Ingenieurs (Dr.-Ing.)

genehmigte Dissertation

von:	Dipl.-Ing. Patrick Diekhake
aus:	Georgsmarienhütte

eingereicht am:	05.11.2015
mündliche Prüfung am:	10.05.2016

Gutachter:	Prof. Dr.-Ing. Dr. h.c. mult. Eckehard Schnieder
	Prof. Dr.-Ing. Jörn-Uwe Varchmin

2016

Vorwort

Die vorliegende Dissertation entstand im Rahmen meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Institut für Verkehrssicherheit und Automatisierungstechnik der Technischen Universität Braunschweig. Die lange Tradition des Institutes, komplexe Aufgaben durch transdisziplinäre und methodische Ansätze zu lösen, stellt die Grundlage diese Arbeit dar und trug letztlich zu deren Gelingen bei. Ich möchte mich an dieser Stelle bei allen bedanken, die mich dabei unterstützt haben.

Mein besonderer Dank gilt meinem Doktorvater Herrn Prof. Dr.-Ing. Dr. h.c. mult. Eckehard Schnieder, dem ehemaligen Leiter des Institutes für Verkehrssicherheit und Automatisierungstechnik der Technischen Universität Braunschweig. Als Forderer und Förderer hat er durch seine Anregungen, Erfahrungen und Ideen die Entstehung der vorliegenden Arbeit maßgeblich beeinflusst.

Herrn Prof. Dr.-Ing. Jörn-Uwe Varchmin, dem ehemaligen Leiter des Instituts für Elektrische Messtechnik und Grundlagen der Elektrotechnik der Technischen Universität Braunschweig, danke ich herzlich für die Übernahme des Koreferates und die spannenden Einblicke in das dieser Dissertation zugrundeliegende Anwendungsgebiet.

Ich danke Herrn Prof. Dr.-Ing. Christoph Herrmann - Leiter des Instituts für Werkzeugmaschinen und Fertigungstechnik der Technischen Universität Braunschweig - für die Übernahme des Vorsitzes der Promotionskommission und seine Ratschläge aus dem Vorgespräch zur Prüfung.

Gute Anregungen und interessante fachliche Diskussionen verdanke ich meinen Kollegen und Freunden. An dieser Stelle sei insbesondere Herrn Dr. Ing. Harald Schrom für die detaillierte Einführung in das der vorliegenden Arbeit zugrundeliegende Automatisierungssystem gedankt. Durch die tatkräftige Unterstützung von ihm und allen Mitarbeitern des Projektes future:workspace erfolgte letztlich die reale Umsetzung des in der Dissertation betrachteten Anwendungsbeispiels.

Nicht zuletzt danke ich meinen Eltern und Geschwistern, die mir in all den Jahren stets den notwenigen Halt gegeben haben. Ich danke meiner Freundin Henrike für den Beistand, ihre liebevolle Unterstützung und die kritische Begutachtung meiner wissenschaftlichen Beiträge.

Braunschweig, Mai 2016 Patrick Diekhake

Kurzfassung

Aufgrund der verbesserten Leistungsfähigkeit und der sinkenden Kosten von Kleinrechnern werden Steuerungseinheiten heute überwiegend dezentral in Verbünden organisiert. Zwar bieten solche verteilten Systeme die notwendige Flexibilität, um die steigenden Anforderungen - z. B. einen erhöhten Funktionsumfang - erfüllen zu können, sie bringen aber auch viele Herausforderungen mit sich. Während der Entwicklungsprozess verteilter Automatisierungssysteme mittels einer durchgängigen Werkzeugkette unterstützt wird, erfolgen die Analysetätigkeiten zumeist isoliert und fokussiert auf eine konkrete Problemstellung innerhalb einer Lebenszyklusphase.

Schwerpunkt der vorliegenden Arbeit ist daher eine Vorgehensweise zur systematischen und vereinheitlichten Analyse von verteilten Automatisierungssystemen. Die Anwendung der erarbeiteten Methodik erfolgt mittels strukturierter Modellierung und Modellausführung am Beispiel eines verteilten Gebäudeautomatisierungssystems.

Ein wesentlicher Bestandteil des Vorgehens ist die Bereitstellung einer verständlichen und gleichzeitig detaillierten Beschreibung des zu analysierenden Systems über ein Systemmodell. Das Modell dient der Wissensrepräsentation und stellt die Basis für nachfolgende Analysen zur Ermittlung (nicht-)funktionaler Eigenschaften dar, indem es das Fundament weiterführender Analysemodelle skizziert. Die Analysemodelle werden im weiteren Verlauf der Arbeit vorgestellt und mathematisch beschrieben. Die nach deren Ausführung der Modelle gewonnenen Analyseergebnisse werden dargestellt und zur Wissensanreicherung in das wissensbeschreibende Systemmodell zurückgeführt.

Nichtfunktionale Eigenschaften wurden u. a. mit Hilfe von Simulationsmodellen am Beispiel folgender Problemstellungen aus der Entwurfsphase analysiert:

- Eine mögliche Degradation der Signalqualität eines maximal ausgedehnten Bitübertragungssystems wurde untersucht und bewertet.
- Die durch physikalisch bedingte Kommunikationsprozesse zwischen verteilten Gerätekomponenten und durch Zyklusübergänge zwischen verteilten Programmkomponenten verursachten Verzögerungszeiten wurden bestimmt.

Weiterhin wurden für zwei Beispielsysteme die Eigenschaften der Beobachtbarkeit und der Steuerbarkeit ermittelt, wodurch die Grundlage für eine abschließende Online-Analyse geschaffen wurde. Eine Petrinetzumgebung, die eine direkte Anbindung an ein Realsystem erlaubt und damit eine Testausführung während der Inbetriebnahme bzw. eine Laufzeitanalyse im operativen Betrieb des Systems ermöglicht, wird vorgestellt.

Inhaltsverzeichnis

Vorwort V
Kurzfassung VII
Verzeichnis der Abkürzungen und Akronyme XI
Symbolverzeichnis XIII
1 Einleitung 1
1.1 Anwendungsfelder verteilter Automatisierungssysteme 2
1.1.1 Anwendungsbeispiel 4
1.2 Probleme und Lösungsansätze verteilter (Gebäude-)Automatisierungssysteme 11
1.3 Analysen im Lebenszyklus eines Automatisierungssystems 17
1.4 Zielsetzung 20
1.5 Aufbau der Arbeit 22
2 Ziel und Konzept einer vereinheitlichten Analyse 24
2.1 Anforderungen 24
2.2 Konzept und Methoden 26
2.2.1 Beschreibungsmittel 28
2.2.2 Werkzeuge 33
3 Systemmodelle zur Wissensrepräsentation 35
3.1 Systemtheoretische Beschreibung 36
3.1.1 Zentrale Systemeigenschaften 36
3.1.2 Begriffsformalisierung und Konkretisierung des Systembegriffs 37
3.1.3 (De-)Komposition von Subsystemen 40
3.2 Gerätetechnische Beschreibung 49
3.3 Anforderungsdefinition 51
3.4 Zusammenfassende Darstellung 54
4 Modellierung und Analyse der Signalqualität und der elektromagnetischen Verträglichkeit 58
4.1 Signalqualität 60
4.1.1 Systemmodell 61
4.1.2 Deskriptive Modelle 64
4.1.3 Analytische Modelle 67
4.1.4 Validierung 70
4.1.5 Einfluss der Teilnehmerzahl 73
4.1.6 Einfluss räumlicher Ausdehnungen 76
4.1.7 Einfluss von Stichleitungen 77
4.2 Störfestigkeit 78
4.3 Störaussendung 80

4.3.1 Einfluss von Filter und Schirmmaßnahmen 80
4.3.2 Einfluss der Signalflankensteilheit 81
4.4 Rückführung der Analyseergebnisse in das Systemmodell 84
5 Modellierung und Analyse von Verzögerungs- und Laufzeiten 86
5.1 Buslastverhalten 87
5.1.1 Systemmodell 87
5.1.2 Exemplarisches Simulationsmodell 91
5.1.3 Validierung 93
5.1.4 Einfluss der Busauslastung 94
5.1.5 Einfluss der Versandart 96
5.1.6 Einfluss der Geräteanzahl 97
5.2 Zyklusübergangs- und Synchronisationsverhalten 98
5.2.1 Systemmodell 98
5.2.2 Simulationsmodell 101
5.2.3 Validierung 103
5.2.4 Simulation eines parallelen Verbundes 106
5.3 Laufzeitermittlung eines kausalen Prozessablaufes 111
5.3.1 Systemmodell 111
5.3.2 Simulationsmodell 116
5.4 Rückführung der Analyseergebnisse in das Systemmodell 122
6 Beobachtbarkeits- und Steuerbarkeitsanalyse 124
6.1 Zustandsdetektierbarkeit am Beispiel einer verarbeitenden Funktion 128
6.2 Zustandssteuerbarkeit am Beispiel einer verarbeitenden Funktion 131
6.3 Sequenzdetektierbarkeit am Beispiel eines kausalen Prozessablaufes 133
7 Analysen im Online-Betrieb 140
7.1 Umsetzung einer Petrinetzumgebung zur Online-Analyse 141
7.1.1 Telegrammübergabe 141
7.1.2 Analysemodelle 142
7.1.3 Gekoppelte Analyseumgebung 142
7.2 Modellbeispiele für die Online-Analyse 143
7.2.1 Modell zur Ausführung eines Testprozesses 145
7.2.2 Modell zur Laufzeitüberwachung 151
8 Zusammenfassung und Ausblick 162
8.1 Zusammenfassung 162
8.2 Ausblick 164

9 Literaturverzeichnis 166
Abbildungsverzeichnis 183
Tabellenverzeichnis 188
Anhang A 189
Anhang B 201

Verzeichnis der Abkürzungen und Akronyme

AAL	Ambient Assisted Living
acatech	Deutsche Akademie der Technikwissenschaften
ANMUL	SmallCAN-spezifische Hardwareressource für analoge und multifunktionale Anwendungen
API	Application Programming Interface
AUSLG	SmallCAN-spezifische Softwareressource für eine Außenluftregler-Anwendung
AUTOSAR	AUTomotive Open System ARchitecture
BACnet	Building Automation and Control Networks
BMW	Beschreibungsmittel, Methoden, Werkzeuge
C2C	Car to Car
CAN	Controller Area Network
CO2	Kohlendioxid
COM	Component Object Model
CORBA	Common Object Request Broker Architecture, Common Object Request Broker Architecture
CPN	Coloured Petri nets
CPS	Cyber-Physical Systems
CSMA/CA	Carrier Sense Multiple Access/Collision Avoidance
DCF	Rufzeichen-Präfix einiger im Langwellenbereich betriebener Sender
DGL	Differenzialgleichung
EEBus	Anwendungsneutrale Schnittstelle für bestehende Kommunikationsstandards
EEG	Erneuerbare-Energien-Gesetz
EHS	European Home System
EIB	Europäische Installationsbus
EMV	Elektromagnetische Verträglichkeit
EnEV	Energieeinsparverordnung
EU	Europäische Union

FlexRay	Serielles, deterministisches und fehlertolerantes Feldbussystem
GA	Gebäudeautomation
GND	Chassis ground
HAN	Home Area Network
HW	Hardware
IPN	Interpretiertes Petrinetz
IT	Informationstechnik
K/I	Kanal/Instanzen
KNX	Feldbus zur Gebäudeautomation
LAN	Local Area Network
LON	Local Operating Network
LT	Linear Technology
MLPS	Multi-Label Protocol Switching
NI	National Instruments
OSI	Open Systems Interconnection
PN	Petrinetz
PROFINET	Process Field Network
RA	Raumautomation
RS485	Norm für bidirektionale differentielle serielle Datenübertragung
SNR	Störabstand
SPICE	Simulation Program with Integrated Circuit Emphasis
SysML	System Modeling Language
TCP	Transmission Control Protocol
TDL	Timing Definition Language
TEM	Transverse Electromagnetic
TTCN	Testing and Test Control Notation
TVS	Transient Voltage Suppressor
UML	Unified Modeling Language
UTP	Unified Modeling Language Testing Profile
VDI	Verein Deutscher Ingenieure
WBS	Wissensbasiertes System

Symbolverzeichnis

A	Querschnittsfläche $[\mathrm{mm}^2]$
$\boldsymbol{A}$	Systemmatrix
$\underline{\underline{\boldsymbol{A}}}$	Matrix mit hinterlegten Aufrufverhalten
b	Datenrate $\left[\frac{\mathrm{bit}}{\mathrm{s}}\right]$
$\boldsymbol{B}$	Steuermatrix
c	Lichtgeschwindigkeit $=3{\cdot}10^8\,\frac{\mathrm{m}}{\mathrm{s}}$
	Konstante
C	Kapazität [F]
$\boldsymbol{C}$	Beobachtungsmatrix
C'	Kapazitätsbelag $\left[\frac{\mathrm{pF}}{\mathrm{m}}\right]$
$\underline{\underline{\boldsymbol{C}}}$	Inzidenzmatrix eines Petrinetzes
$\underline{\underline{\boldsymbol{D}}}$	Matrix zur Zuordnung von Daten
$\underline{\boldsymbol{d}}$	Vektor zur Zuordnung von Daten
D	Anzahl an Daten in einem System
$\underline{\underline{\boldsymbol{E}}}$	Matrix zur Zuordnung von Eingängen
f	Zustandsübergangsfunktion
F	Anzahl an Funktionen in einem System
$\boldsymbol{F}$	Zustandsübergangsmatrix
$\underline{\boldsymbol{f}}$	Vektor zur Zuordnung von Funktionen
g	Zählvariable
G	Leitwert $[S]$
	Anzahl an Geräten in einem System
$\underline{\underline{\boldsymbol{G}}}$	Matrix zur Zuordnung von Geräten
$\underline{\boldsymbol{g}}$	Vektor zur Zuordnung von Geräten
Gr	Anzahl sequenzdetektierbarer Gruppen
$\underline{\underline{\boldsymbol{Gr}}}$	Matrix zur Zuordnung von sequenzdetektierbaren Gruppen
h	Ausgabefunktion
$\boldsymbol{H}$	Ausgabematrix
i	Zählvariable
	Zeilen in einer Matrix

I	Stromstärke [A]
$\underline{\underline{\boldsymbol{I}}}$	Einheitsmatrix
j	Imaginäre Zahl
j	Zählvariable
	Spalten in einer Matrix
k	Diskreter Zeitpunkt
	Zählvariable
	Parameter der Poissonverteilung
K	Anzahl an Kanten in einem Netz
l	Leitungslänge [m]
L	Induktivität [H]
L'	Induktivitätsbelag $\left[\frac{\mathrm{H}}{\mathrm{m}}\right]$
n	Maximale Anzahl an Busteilnehmern in einem System
	Ordnung eines Systems
N	Anzahl an Knoten in einem Netz
	Zählvariable
$\boldsymbol{N}$	Inzidenzmatrix eines Petrinetzes
$\underline{\boldsymbol{o}}$	Vektor mit hinterlegten Prioritätswerten
P	Anzahl an Programmkomponenten in einem System
$\underline{\underline{\boldsymbol{P}}}$	Matrix zur Zuordnung von Programmen
$\underline{\boldsymbol{p}}$	Vektor zur Zuordnung von Programmen
Pr	Anzahl an Prozessschritten
R	Widerstand [Ω]
R'	Widerstandsbelag $\left[\frac{\Omega}{\mathrm{m}}\right]$
s	Stelle in einem Petrinetz
S	Menge von Stellen in einem Petrinetz
$\underline{\underline{\boldsymbol{S}}}$	Matrix zur Zuordnung von Ausgängen
t	Zeit [s]
	Zeitparameter [s]
t	Transition in einem Petrinetz
T	Zeitspanne [s]
T	Menge von Transitionen eines Petrinetzes

$\underline{\boldsymbol{t}}$	Vektor mit hinterlegten Zeitwerten
$\underline{\underline{\boldsymbol{Ts}}}$	Matrix zur Zuordnung von sichtbaren Funktionen
u	Eingangsgröße eines Systems / Automaten / Petrinetzes
U	Spannung [V]
v	Geschwindigkeit $\left[\frac{\mathrm{m}}{\mathrm{s}}\right]$
v	Ausgangsgröße eines Automaten /Petrinetzes
V	Verteilungsfunktion
x	Räumliche Variable [m]
	Zustandsvariable eines Systems
x	Element einer Menge
y	Ausgangsgröße eines Systems
z	Zustand eines Automaten
γ	Leitungsdämpfung
λ	Parameter der Poissonverteilung
μ	Präfix 10^{-6}
	Parameter der Lognormalverteilung
ρ	Spezifischer Widerstand $\left[\frac{\Omega\ \mathrm{mm}^2}{\mathrm{m}}\right]$
σ	Parameter der Lognormalverteilung
τ	Zeitkonstante [s]
$\underline{\underline{\boldsymbol{\varphi}}}$	Matrix zur Zuordnung von messbaren Stellen
ω	Kreisfrequenz $\left[\frac{1}{\mathrm{s}}\right]$
ΔU	Spannungsdifferenz [V]
Δt	Zeitdifferenz [s]
ΔT	Menge von Zeitdifferenzwerten

1 Einleitung

> "At scale, everything breaks … Keeping things simple and yet scalable is actually the biggest challenge … Most things don't work that well at scale."

So äußerte sich Urs Hölzle, Senior Vice President für die technische Infrastruktur bei Google, zu den Problemen, die bei zunehmender Größe und Vernetzung von Systemen zu erwarten sind. Die drei Abschnitte des Zitates lassen sich wie folgt interpretieren.

- Der Anwendungsbezug sowie die zu erwartenden Probleme werden allgemeingültig und unspezifisch formuliert. Demnach ist in allen Anwendungsfeldern, in denen *„Dinge"* hochskaliert werden, mit einer Degradation von Systemeigenschaften bis hin zu einem Totalausfall des Systems zu rechnen. Dieser Umstand ist insbesondere als bedeutsam und gewichtig einzustufen, wenn verteilte Automatisierungssysteme, für die zumeist eine hohe Zuverlässigkeit gefordert wird, als hochskalierte Systeme betrachtet werden.

- Im Gegensatz zu den Anwendungsfeldern und den zu erwartenden Problemen wird die sich ergebene Herausforderung eindeutiger dargelegt: Ziel muss es sein die *„Dinge"* möglichst einfach und dennoch skalierbar zu halten. Das bedeutet, dass die Komplexität eines Systems weniger durch deren Vielfalt sondern vielmehr durch die Anzahl seiner *„Dinge"* charakterisiert sein sollte. Dabei ist der Begriff *„Ding"* wiederum als ein unspezifischer Gegenstand beschrieben, welcher als Bestandteil des Systems selbst aber auch als Bestandteil einer dem System zugrundeliegenden Analyseaufgabe aufgefasst werden kann.

- Der Analyse, d. h. der Ermittlung noch unbekannter Eigenschaften eines Systems, ist insbesondere dann ein hoher Stellenwert beizumessen, wenn die am Ende des Zitates verankerte Aussage zum Verhalten hochskalierter Systeme mit in die Interpretation einfließt. Demnach ist bei der Analyse von großen Systemen davon auszugehen, dass für diese nur noch vage Abschätzungen hinsichtlich ihrer Eigenschaften und deren Auswirkungen angegeben werden können.

Nur unter Berücksichtigung der interpretierten Herausforderung, dass das System selbst aber auch dessen Analyse den Anforderungen an Einfachheit und Skalierbarkeit gerecht werden, bleibt die Möglichkeit erhalten, unsicheren Annahmen und vagen Beurteilungen konkrete und präzise Aussagen entgegenstellen zu können. Daher wird in der vorliegenden Arbeit die Analyse verteilter Automatisierungssysteme systematisch behandelt. Eine Vereinheitlichung von Analyseaufgaben stellt dabei eine

wesentliche Voraussetzung dar, um Systemeigenschaften komplexer und hochskalierter Automatisierungssysteme umfassend und eindeutig charakterisieren bzw. beurteilen zu können.

In den nachfolgenden einleitenden Abschnitten werden die Motivation und die Herausforderungen einer vereinheitlichten Analyse tiefgreifender verdeutlicht. Dazu wird in Abschnitt 1.1 zunächst auf das breite Spektrum möglicher Anwendungsfelder verteilter Automatisierungssysteme eingegangen. Aus dem Bereich der Gebäude- bzw. Raumautomation wird eine realisierte Anwendung im Detail vorgestellt, auf die sich die im weiteren Verlauf der Arbeit behandelten Analysebeispiele beziehen. Zur Darlegung der Herausforderungen werden in Abschnitt 1.2 die unterschiedlichen Problemstellungen von verteilten (Gebäude-)Automatisierungssystemen aufgezeigt sowie in Abschnitt 1.3 die Vielfalt vorhandener Analyseverfahren erläutert. Die Zielsetzung und die sich ergebene Struktur der Arbeit werden in den abschließenden Abschnitten dieses Kapitels vorgestellt.

1.1 Anwendungsfelder verteilter Automatisierungssysteme

Verteilte Automatisierungssysteme weisen hinsichtlich der Erfüllung steigender Anforderungen viele Vorteile auf. So können beispielsweise durch Nebenläufigkeit Rechenprozesse optimiert und die Fehlertoleranz erhöht werden. Darüber hinaus sind verteilte Systeme gegenüber zentralisierten Systemen einfacher skalierbar und können so den gestiegenen Anforderungen an Flexibilität und Wiederverwendbarkeit eher gerecht werden. Dezentralisierte Systeme kommen daher in vielen Anwendungsgebieten der Automatisierungstechnik zum Einsatz.

In der *Industrieautomation* stellen Betreiber moderner Produktionsanlagen Anforderungen hinsichtlich Flexibilität und Wiederverwendbarkeit von Hard- und Softwarekomponenten an die Hersteller industrieller Automationskomponenten. Daher wurde mit der Einführung der [DIN EN 61499-1] ein generisches Modell für verteilte Steuerungssysteme auf Basis einer objektorientierten Beschreibung definiert, um die Entwicklung verteilter Steuerungsapplikationen zu unterstützen. Ein weiteres Beispiel für die Dezentralisierung von Automatisierungssystemen zeigt die Realisierung einer echtzeitfähigen Kommunikationsarchitektur, die in [Dadji et al. 2010] vorgestellt wurde und für eine verteilte PC-basierte Steuerung von Industrierobotern eingesetzt werden kann. Es zeigt sich also, dass in der Industrieautomation die zentral organisierten Automatisierungsstrukturen verstärkt durch dezentrale Konzepte ersetzt werden [Fay et al. 2008]. Basierend auf neueren Vorstellungen zu Industrie 4.0 erfolgt die ganzheitliche Steuerung einer zukünftigen Industriestätte auf einer dezentralisierten

Ebene und über autonom agierende leistungsfähige Kleinstcomputer. Solche Cyber Physical Systems (CPS) vernetzen zukünftig Vertrieb, Produktion und Logistik, um sowohl Massenartikel als auch individuelle Einzelstücke effizienter produzieren zu können [acatech 2013].

In der *Fahrzeugautomatisierung* haben verteilte Automatisierungssysteme schon früh Einzug erhalten, da eine diskrete Verkabelung von Sensoren und Aktoren an zentralisierte Steuergeräte zu einer schlechten Handhabbarkeit aufgrund des sich ergebenden Gewichtes und der Geometrie des Bordnetzes führte [Zimmermann/Schmidgall 2006]. Daher sind in den 90er Jahren die Netzwerkumfänge auch in unterklassigen Fahrzeugen stark gestiegen, so dass in heutigen Fahrzeugen bereits zahlreiche Steuergeräte vernetzt sind, die über verschiedene Bussysteme miteinander kommunizieren [Gevatter/Grünhaupt 2004: 14]. Um eine hohe Variantenvielfalt im Funktionsumfang anbieten zu können, setzen Fahrzeughersteller auch auf dezentralisierte Konzepte mit wiederverwendbaren Softwaremodulen. Ein solcher Ansatz zeigt sich insbesondere bei der Entwicklung von Standards für Systemfunktionen, beispielsweise durch die AUTOSAR-Partnerschaft [Schäuffele/Zurawka 2013]. Ziel dieses Zusammenschlusses namhafter Automobilfirmen ist die Erarbeitung von Methoden, die es ermöglichen, dass Softwarekomponenten wiederverwendet, ausgetauscht und skaliert werden können. Für eine zukünftige vollständige Automatisierung von Fahrzeugen zur Steigerung der Verkehrssicherheit sind neben einer vernetzten Umfeldwahrnehmung durch mehrere Fahrerassistenzsysteme auch hochgenaue Ortungseinheiten sowie eine Fahrzeugkommunikation, z. B. Car to Car (C2C) erforderlich [Schnieder 2007]. Aus makroskopischer Sicht stellt damit zukünftig das Fahrzeug selbst eine Komponente innerhalb des übergeordneten Systems zur *Verkehrsautomation* dar.

Ebenso wird im Anwendungsgebiet der *Energieautomation* auf dezentrale Lösungskonzepte gesetzt, um das von der Bundesregierung festgelegte Ziel, den CO_2-Ausstoß bis 2020 um 40 % gegenüber 1990 zu senken, zu erreichen. Im Energienetz werden zunehmend kleinere, dezentrale Erzeugungsanlagen eingesetzt, für die durch das Erneuerbare-Energien-Gesetz [EEG 2014] bereits Anreize geschaffen wurden. Dieser Wandel in der Energieversorgung führt zu einem Übergang von der bisherigen Top-Down Struktur zu einer bidirektionalen Struktur, wodurch sich die Betriebsführung moderner Energienetze grundlegend ändern wird. Den dezentral im Verteilnetz installierten Erzeugungsanlagen werden zunehmend mehr Systemdienstleistungen und Steuerungsaufgaben übergeben, wodurch eine Kommunikation sämtlicher verteilter Konstituenten des Elektrizitätsversorgungsnetzes wie Stromerzeuger, Speicher und elektrische Verbraucher unumgänglich wird [Diekhake et al. 2014].

Zur nachhaltigen Einsparung von Energie sind auch energieeffiziente Gebäude in das Themenfeld der Energieautomation integriert worden, da innerhalb der Europäischen Union heute 40 % des Gesamtverbrauchs an Primärenergie auf den Gebäudesektor entfallen [EU 2010/31] und durch den Einsatz von Systemen für die *Gebäudeautomation* (GA) der Primärenergieverbrauch eines Gebäudes um bis zu 30 % gesenkt werden kann [Fountain et al. 1996]. Richtlinien wie beispielsweise die Energieeinsparverordnung [EnEV 2014] berücksichtigen bei der Ermittlung des Primärenergieverbrauch eines Gebäudes zur Erstellung eines „Energieausweises" erstmals den Einfluss von GA-Systemen. In der Norm [DIN EN 15232] wird deren Einfluss auf die Energieeffizienz von Gebäuden konkreter beschrieben, indem Effizienzklassen für GA-Systeme eingeführt werden. Die energieeffizientesten Klassen werden unter Berücksichtigung einer bedarfsgeregelten Einzelraumregelung erreicht, die durch Funktionen der *Raumautomation* (RA) realisiert wird, auf die die Richtlinie [VDI 3813:2011] im Detail eingeht. Eine Einzelraumregelung erfordert den Einsatz dezentraler Gerätekomponenten innerhalb eines Raumes, wodurch es dazu kommen kann, dass GA-Systeme mehrere 10000 Geräte beinhalten können [Hunstock et al. 2000]. Neben Funktionen zur Steigerung der Energieeffizienz erfüllt die Gebäudeautomation zusätzlich weitere vielfältige Aufgaben aus den Bereichen Sicherheit und Komfort, wodurch der Funktions- und Geräteumfang weiter ansteigt. Als Beispiel seien hier Funktionen für das Ambient Assisted Living (AAL) zu nennen, die notwendig sind, um dem Bedürfnis älterer oder gesundheitlich beeinträchtigter Menschen nach einem eigenständigen und unabhängigen Leben in ihrer gewohnten Umgebung nachkommen zu können [Brackmann/Varchmin 1996], [Georgieff 2008]. Solche AAL-Systeme übernehmen u. a. Überwachungsfunktionen oder Funktionen zur Notrufauslösung. Zusammenfassend ist auch ein GA-System ein großes, verteiltes und sukzessiv erweiterbares System mit mehreren gewerkeübergreifend agierenden Geräten und Programmen, das der Überwachung, Regelung, Steuerung und Optimierung von Energie, Komfort und Sicherheit dient.

1.1.1 Anwendungsbeispiel

Im Folgenden wird ein Anwendungsbeispiel für ein verteiltes Automatisierungssystem vorgestellt. Es handelt sich um die Implementierung eines Systems zur Raumsteuerung eines Büros, in dem ca. 100 Gerätekomponenten installiert wurden [Eickmeyer et al. 2012], [Schrom et al. 2011], [Diekhake 2014]. Die in diesem Demonstrationsobjekt umgesetzten Funktionen (Raumklima, Sonnenschutz, Beleuchtung) erfüllen die Anforderungen der höchsten Energieklasse nach DIN EN 15232. Das aufgeführte Beispiel beschränkt sich lediglich auf die Betrachtung von 44

RA-Funktionen zur Regulierung des Raumklimas, die durch 30 Geräte und 34 Programmkomponenten umgesetzt wurden.

Tabelle 1: Zuordnung von SmallCAN-Ressourcen zu Funktionen aus [VDI 3813:2011]

		Programm-Nr.	1	3	5	7, 14, 15	8	9	11	2, 19	6	4	10	13	12	16	17	18	21	24	22	25, 28	20,23	26, 27, 29, 30, 31, 32, 33, 34
		Geräte-Nr.	1	3	4	6, 12, 13	7	8	10	2, 17	5	3	9	11	10	15	16	14	18	20	19	21, 24	18, 20	22, 23, 25, 26, 27, 28, 29, 30
		(Anwendungs-) Programm / Modul	DCF	PIR	STD	TEMPF	TEMPM	FEUCH	CO2	4TAST	PSF	PRAES	SOLLT	KLIMR	LUFTR	HKOOR	KKOOR	LKOOR	AUSLG	UMLUG	ZENLG	DCOUT	ANMUL	MPBUS
		Gerät /Programm	Funkuhr	Bewegungsmelder	Fensterkontakt	Temperaturfühler	Raumthermostat	Feuchtigkeitssens	CO_2-Sensor	Schalter / Taster	Kalender	Belegungsauswert	Sollwertermittler	Temperaturregler	Lüftungsregler	Heizkoordinator	Kühlkoordinator	Lüftungskoordinat	Außenluftgregler	Umluftregler	Zentralluftregler	Spannungsausgan	Analogmodul	Stellantrieb
RA-Funktionen nach VDI 3813		**SmallCAN-Ressourcen**																						
	Fkt.-Nr.	**Ressourcenzuweisung**																						
Sensorfunktion	1		**X**	**X**	**X**	**X**	**X**	**X**	**X**															
Präsenzerkennung	3			X																				
Fensterüberwachung	5				X																			
Temperaturmessung	8,10,17, 18					X	X																	
Taupunktüberwachung	11							X																
Luftqualitätsmessung	14								X															
Bedien-/Anzeigefunktion							**X**			**X**	**X**													
Temperatursollwert stellen	7,9						X				X													
Raumnutzung wählen	27									X														
Präsenz melden	2									X														
Anwendungsfunktionen											**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**			
Zeitprogramm	6										X													
Belegungsauswertung	4											X												
Energieauswahl	12												X											
Sollwertermittlung	13												X											
Funktionswahl													X											
Temperaturregelung	16													X										
Luftqualitätssteuerung	15														X									
Stellwertbegrenzung	19-26															X	X	X						
Sequenzsteuerung	31																		X					
Frostschutz	31																		X					
Raum-Zulufttemperatur-Regelung	31, 34																		X	X				
Steuerung über Raumnutzungsart	31, 34, 32																		X	X	X			
Volumenstromregelung	34, 32																				X			
Aktorfunktion	28,35,38																					**X**	**X**	**X**
Stellantriebsaktor	29,30,33, 36,37,39 40,41,42, 43,44																						X	X

Der Raum ist u. a. mit mehreren Heiz-, Kühl- und Lüftungssystemen ausgestattet worden, die individuell gewichtet die Temperatur- und Luftqualitätsregelung unter Berücksichtigung von CO_2-Gehalt, Präsenzinformationen, Außentemperatur, gewünschten Betreiber- und Nutzereinstellungen, Zeitangaben und Behaglichkeitskurven realisieren. In Tabelle 1 sind die für den Büroraum entwickelten Ressourcen, d. h. Geräte- und Programmkomponenten den zugrundeliegenden Raumautomatisierungsfunktionen aus der Richtlinie [VDI 3813:2011] gegenübergestellt. In Abbildung 1.1 sind die Wirkzusammenhänge der aufgeführten Automatisierungsanlagenbestandteile über die Funktionen der Raumautomation dargestellt. Weiterhin beinhaltet die Abbildung die Beschreibung der Systemarchitektur gemäß Tabelle 1, d. h. die Zuordnung der Funktionen zu den Programm- und Gerätekomponenten. Einige Funktionen sind ggf. in einer Makrofunktion zusammengefasst wie beispielsweise die Außenluftregelung (Fkt.-Nr. 31) bestehend aus Sequenzsteuerung, Frostschutz, Raumzulufttemperaturregelung und Raumnutzungsartsteuerung.

Im Folgenden werden die Wirkzusammenhänge für den Büroraum erläutert. Über eine Funkuhr (DCF) wird die aktuelle Uhrzeit ZEIT an das Zeitprogramm (PSF) übergeben, welches entsprechend eine zentrale Energievorgabe M_BMS ausgibt. Ergänzend wird über einen Fensterkontakt (STD) der Öffnungszustand des Fensters B_WINDOW sowie der aktuelle Anwesenheitsstatus P_ACT an die Energieniveauauswahl-Funktion in Programm zur Sollwertermittlung (SOLLT) übermittelt. Der aktuelle Anwesenheitsstatus P_ACT wird aus den Präsenzzuständen P_AUTO des Bewegungssensors (PIR) und P_MAN des manuell bedienbaren Schalters (4TAST) mittels des Belegungsauswerters (PRAES) bestimmt. Über die Sollwertermittlung wird aus dem aktuellen Energieniveau M_ACT, der Vorgabe eines Temperatursollwertes durch den Betreiber T_BMS, die durch den Außenfühler (TEMPF) gemessene Außentemperatur T_OUT sowie der über ein Raumthermostat (TEMPM) in gewissen Stufen von Nutzer einstellbare Sollwertkorrektur T_SETPT ein optimierter Temperatursollwert T_SETPTS errechnet. Über die übergeordnete Anwendungsfunktion (Makrofunktion mit Fkt.-Nr. 13) wird außerdem die aktuelle Regelfunktion F_ACT in Abhängigkeit der Taupunktnähe B_DEW und der Raumtemperatur T_ROOM ermittelt. Der Temperaturregler (KLIMR) fordert entsprechend seiner Eingangsgrößen eine Heiz- bzw. Kühlleistung X_SET_HEIZ/KUEHL für den Raum an. Ein parametrierbarer Koordinator zum Heizen (HKOOR) bzw. zum Kühlen (KKOOR) teilt die angeforderte Heiz- bzw. Kühlleistung in Teilleistungen X_SET_HEIZ1…8 bzw. X_SET_KUEHL1…8 auf und stellt diese verschiedenen Klimagerätefunktionen zur Verfügung. Dazu zählen u. a. ein Außenluft-

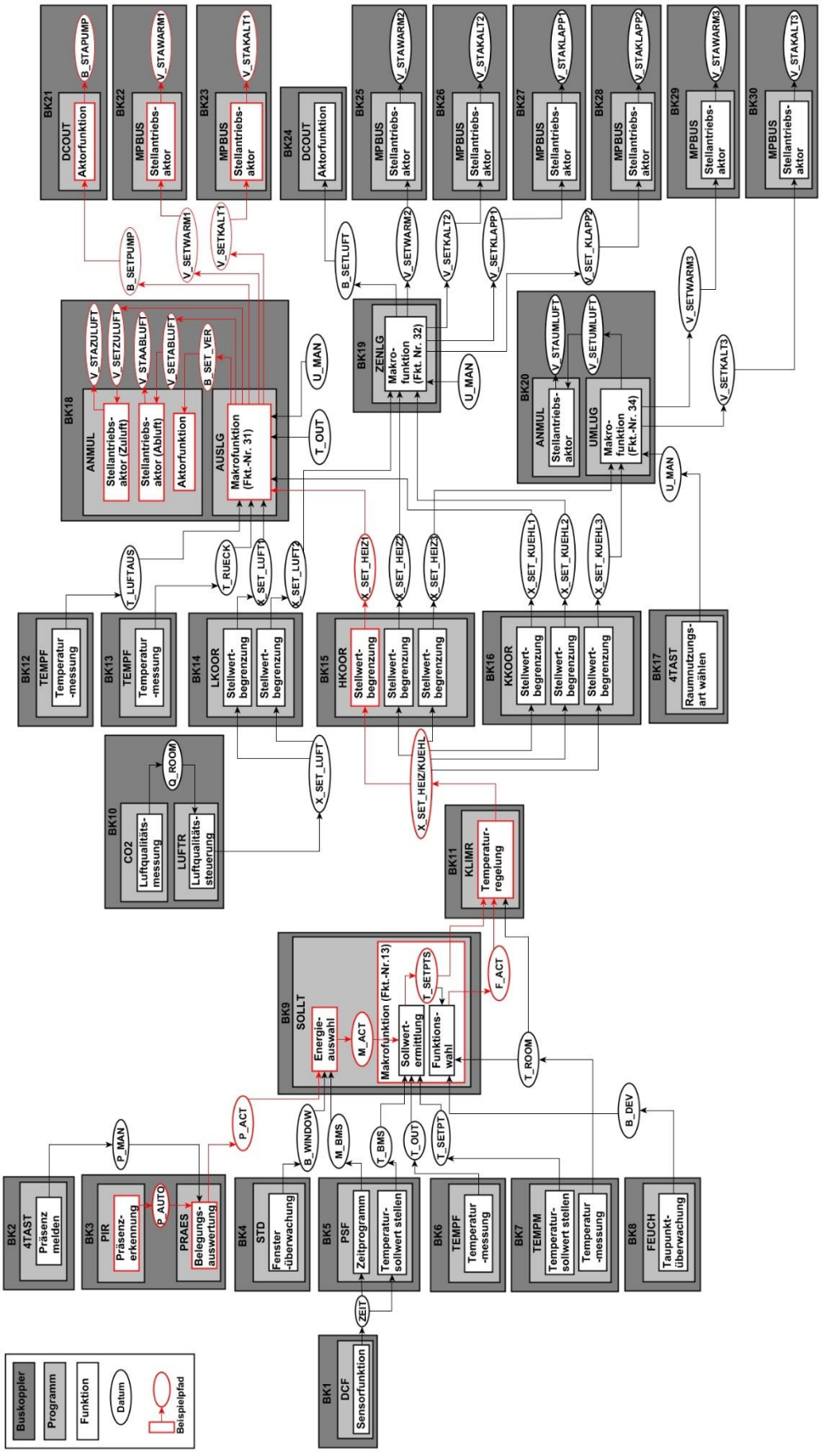


Abbildung 1.1: Ausschnitt aus der manuell erstellten Systemarchitektur eines durch SmallCAN gesteuerten Büroraumes

gerät (AUSLG), ein zentrales Lüftungsgerät (ZENLG) sowie ein Umluftgerät (UMLUG). Für eine kooperative Lüftung wird dem Lüftungskoordinator (LKOOR) ein Stellwert X_SET_LUFT übergeben, der über einen Lüftungsregler (LUFTR) in Abhängigkeit des aktuellen CO_2-Wertes Q_ROOM bestimmt wird (CO2). Der Koordinator ermittelt die Teilluftwechselraten X_SET_LUFT1…8, die von den an der Luftqualitätssteuerung beteiligten Klimagerätefunktionen empfangen werden. In dieser Beispielanwendung erfolgt die kooperative Lüftungsregelung über das Außenluftgerät (AUSLG) und das zentrale Lüftungsgerät (ZENLG). Das Außenluftgerät (AUSLG) steuert eine Pumpe (DCOUT) über B_SET_PUMP, gibt die Stellwerte V_SETWARM und V_SETKALT für die Stellventile (MPBUS) vor, stellt die Stromversorgung der Lüftergeräte (ANMUL) über das Datum B_SET_VER her und führt ihnen ihre Stellwerte V_SETZULUFT und V_SETABLUFT zu. Zur optimalen Steuerung und Regelung (z. B. Frostschutz oder Raum-Zulufttemperatur-Regelung) liest die Anwendungsfunktion des Außenluftgerätes außerdem die manuell vom Nutzer eingestellte Raumnutzungsart U_MAN, die Luftauswurftemperatur T_LUFTAUS, die Rücklauftemperatur des Warmwasserkreises T_RUECK und die gemessene Außentemperatur T_OUT ein.

Belegungsauswertung

Exemplarisch wird die Funktion zur Belegungsauswertung (Fkt.-Nr. 4), die durch die Programmkomponenten PRAES realisiert ist, im Detail erläutert. Zur graphischen Beschreibung von Steuerungsaufgaben, d. h. einzelner Verarbeitungsfunktionen aber auch ganzer Anlagenfunktionen empfiehlt die Richtlinie [VDI 3814:2009] die Anwendung von Zustandsgraphen. Für deren Erstellung werden die benötigten Datenpunkte in einer Zuordnungstabelle zusammengefasst, auf deren Basis sich dann die Zustandsgraphen ableiten lassen. In Tabelle 2 ist die Zuordnungstabelle und in Abbildung 1.2 der Zustandsgraph sowie das Realverhalten der Belegungsauswertung dargestellt. Die Funktion zur Belegungsauswertung ermittelt für alle nachfolgenden Anwendungsfunktionen den aktuellen Belegungszustand innerhalb eines Raumes [VDI 3813:2011]. Für die verschiedenen Möglichkeiten der An- oder Abmeldung werden die Ausgabeinformationen der *Präsenzerkennung* (Fkt.-Nr. 3) und der Bedienfunktion *Präsenz melden* (Fkt.-Nr. 2) ausgewertet. Die einfachste Realisierung der Belegungsauswertung entspräche einer Oder-Verknüpfung beider Eingänge P_AUTO und P_MAN, die direkt zu einem binären Belegungszustand am Ausgang P_ACT führt. Die umgesetzte Lösung nach Abbildung 1.2 hingegen weist sowohl für den Eingang P_AUTO als auch für den Ausgang P_ACT einen erweiterten Wertebereich auf. Für die Eingangsinformationen aus der Präsenzerkennung wird zwischen einem Bewegungsalarm P_AUTO-1 als Impuls und einem sich aus diesem

ergebenen Präsenzzustand P_AUTO-0 als Pegel unterschieden. Der Ausgang P_ACT wird durch die Zustände P_ACT-0 *(Keine Präsenz)*, P_ACT-1 *(Kurze Präsenz)*, P_ACT-2 *(Lange Präsenz)* und P_ACT-4 *(Ablaufwarnung)* attribuiert.

Tabelle 2: Zuordnungstabelle für die Funktion zur Belegungsauswertung

Datenpunktbenennung	**Kurzzeichen**	**Logische Zuordnung**
Eingangsgrößen		
Belegungszustand aus Bedienvorgang	P_MAN	Aus: P_MAN=0
Bewegungsalarm	P_AUTO-1	Alarm: P_AUTO-1 =1
Präsenzzustand durch Präsenzmelder	P_AUTO-0	Präsenz: P_AUTO-0 =1
Ausgangsgrößen		
Keine Präsenz	P_ACT-0	Keine Präsenz: P_ACT-0=1
Kurze Präsenz	P_ACT-1	Kurze Präsenz: P_ACT-1=1
Lange Präsenz	P_ACT-2	Lange Präsenz: P_ACT-2=1
Ablaufwarnung	P_ACT-4	Ablaufwarnung: P_ACT-4=1
Zeitglieder		
Nachlaufzeit kurze Präsenz	Δt1	Δt1=1 min
Ablaufzeit lange Präsenz	Δt2	Δt2=15 min
Dauer bis lange Präsenz	Δt3	Δt3=5 min
Dauer Ablaufwarnung Präsenz	Δt4	Δt4=40 sec
Dauer Ablaufwarnung Verlassen	Δt5	Δt5=1 min

Entsprechend den verschiedenen Ein- und Ausgangsausprägungen umfasst die realisierte Belegungsauswertung im Gegensatz zur Minimallösung eine mannigfaltigere Funktionslogik. Initial ausgehend von einer nicht vorhandenen Präsenz geht die Funktion zur Belegungsauswertung in die kurze Präsenz über, sobald ein Bewegungsalarm und der sich anschließende Präsenzzustand empfangen werden. Sofern innerhalb der Nachlaufzeit der kurzen Präsenz (Δt1) der von der Präsenzerkennung erzeugte Präsenzzustand nicht mehr aktiv ist, geht der Belegungszustand in eine ungeblockte Ablaufwarnung über. Nach der Dauer der ungeblockten Ablaufwarnung (Δt4) erfolgt der Übergang in den Belegungszustand *Keine Präsenz*, es sei denn es wird zwischenzeitlich erneut ein aktiver Präsenzzustand eingelesen, um wieder in die kurze Präsenz zurückzuspringen. Ebenso kann aus der kurzen Präsenz bzw. aus der Ablaufwarnung in den Belegungszustand der langen Präsenz übergegangen werden, sobald die Zeitspanne bis zur lange Präsenz (Δt3), be-

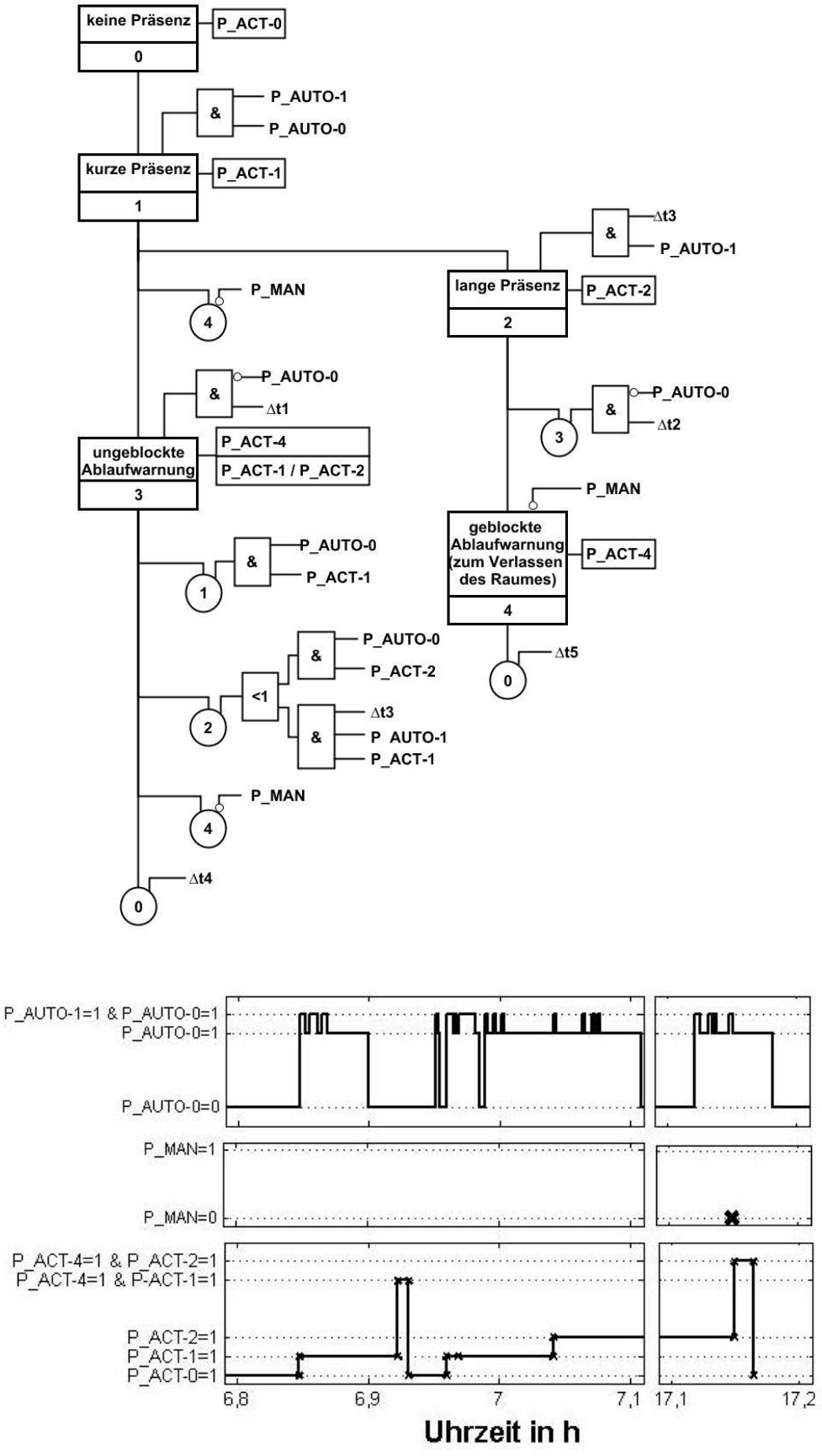


Abbildung 1.2: Zustandsgraph (oben) und Systemverhalten (unten) der Funktion zur Belegungsauswertung

ginnend ab dem Eintritt in den Belegungszustand *Kurze Präsenz*, erreicht ist und anschließend ein Bewegungsalarm detektiert wird. Letztlich kann durch eine manuelle Abmeldung (P_MAN=0) der Zustand der geblockten Ablaufwarnung erreicht werden. Folglich wird durch die der Belegungsauswertung nachfolgenden Anwendungsfunktionen der Raum in einen belegungsfreien Zustand überführt und innerhalb der Dauer der geblockten Ablaufwarnung (Δt5) werden die eingehenden Informationen aus der Präsenzerkennung von der Funktion zur Belegungsauswertung ignoriert. Der Nutzer kann in dieser Zeitspanne den Raum verlassen ohne einen aktiven Belegungszustand auszulösen und sich so über den Zustand nach dem Verlassen des Raumes vergewissern.

1.2 Probleme und Lösungsansätze verteilter (Gebäude-)Automatisierungs-systeme

Der Einsatz verteilter Automatisierungssysteme geht einher mit einer Vielzahl an Herausforderungen [Kurschl 2000: 33]. Im Folgenden werden die wesentlichen Problemstellungen der steigenden *strukturellen Komplexität*, der erhöhten Anzahl an *einflussnehmenden Faktoren*, der Verwendung *gemeinsamer* sowie *verteilter Ressourcen*, des stetig steigenden *Funktionsumfangs*, des Anstiegs *partieller Fehler* sowie der zu berücksichtigenden *Sicherheit* erörtert. Deren Wirkungen auf Systeme der Gebäudeautomatisierung werden diskutiert sowie vorhandene Lösungs- bzw. Analyseansätze vorgestellt.

Bereits am eingeführten Anwendungsbeispiel aus Abschnitt 1.1.1 wird deutlich, dass verteilte Systeme aufgrund der erhöhten Anzahl an verwendeten Komponenten und deren Verknüpfungen im Allgemeinen eine hohe strukturelle *Komplexität* aufweisen. Im Anwendungsgebiet der Gebäudeautomatisierung ist die strukturelle Komplexität durch die Anzahl und Verknüpfungen der Programm- und Gerätekomponenten selbst [Hunstock et al. 2000] aber auch durch deren Vielfalt gekennzeichnet, da ein großes Portfolio an Sensoren, Aktoren und Steuerrechnern von unterschiedlichen Herstellern mit unterschiedlichen Kommunikationsprotokollen, Plattformen und Semantiken angeboten wird. Dieser Umstand führt zu gerätetechnischen Interoperabilitätsproblemen bei der Inbetriebnahme von GA-Systeme [Werthschulte 2003]. Weiterhin ist eine Analyse im laufenden Betrieb nur eingeschränkt möglich, da diese von den verschiedenen Teilsystemen unterschiedlich umfassend unterstützt wird bzw. der Austausch von Messdaten aufgrund eines fehlenden einheitlichen Formates erschwert wird.
Durch eine Reglementierung bei der Entwicklung eines verteilten Automatisierungssystems kann das Interoperabilitätsproblem dahingehend minimiert

werden, dass einheitliche und wiederverwendbare Komponenten zur Anwendung kommen. Dadurch wird die strukturelle Komplexität im Wesentlichen nur noch durch die erhöhte Anzahl an Systemelementen und weniger durch deren Vielfalt beeinflusst. Dieser Entwurfsansatz setzt Konzepte der Standardisierung und Vorgaben für Geräte und Kommunikationsprotokolle voraus. Für die Gebäudeautomation existieren bereits bedeutende gewerkeübergreifende und offene Standards wie BACnet, EEBus, LON, KNX, als Nachfolger der Feldbusse EIB, BatiBus und EHS [Brackmann/Varchmin 1997] oder SmallCAN, wobei BACnet und EEBus keine Bussysteme im herkömmlichen Sinne sind, sondern vielmehr zur Vereinigung existierender Systeme beitragen sollen. Systeme wie KNX haben den Nachteil, dass spezifische Busgeräte teuer sind [Schrom 2003: 68ff], [Bussenius 2014: 44], wodurch die Kosten für eine ganzheitliche Installation hoch ausfallen. Weiter besteht ggf. eine Abhängigkeit vom Installateur, der das System konfiguriert bzw. parametriert. Ebenso sind die Betriebskosten aufgrund des signifikanten Eigenstromverbrauchs als hoch zu bewerten. Schrom hat diese Problematik frühzeitig erkannt und mit dem System SmallCAN eine optimierte Lösung für die ganzheitliche Kommunikation in einem GA-System realisiert [Schrom 2007]. In dem Projekt *SmallCAN* [Diekhake et al. 2011] erfolgte die Weiterentwicklung des Systems für den Einsatz als gewerkeübergreifendes GA-System durch die Schaffung einer Vielfalt und Bandbreite an Applikationsmodulen. Die praktische Anwendung wurde u. a. im Rahmen des Projektes *future:workspace* [Eickmeyer et al. 2012], [Diekhake 2014] (Abschnitt 1.1.1) sowie in den Projekten *DIGAflex* [Diekhake et al. 2014] und *EnEffStadt* [Koch et al. 2014] vollzogen.
Die strukturelle Komplexität kann auch beherrschbar gehalten werden, indem ein gemeinsames Systemverständnis aufgebaut bzw. herbeigeführt wird, u. a. durch die eindeutige Klärung allgemeiner und fachspezifischer Begriffe (Kapitel 3). In [Stein et al. 2010] wird ein Werkzeug vorgestellt, mit dem auf Basis einer eindeutigen Definition von Begriffen und deren Relationen zueinander ein systembeschreibendes Begriffsgebäude erstellt werden kann. In [Runde et al. 2011] werden wissensbasierte Systeme vorgeschlagen, die für eine transparente Darstellung von Systemwissen und für die Analyse eines Systems genutzt werden können. Der Einsatz von Werkzeugen zur Schaffung eines gemeinsamen Systemverständnisses wird u. a. im Anwendungsfeld der Gebäudeautomatisierung in [Runde 2011] am Beispiel eines wissensbasierten Systems (WBS) behandelt.

Die strukturelle Komplexität spiegelt sich auch im Systemverhalten wider, welches durch eine Fülle *einflussnehmender Faktoren* bestimmt wird. Als Beispiel sei die Feldebene der Gebäudeautomatisierung zu nennen, auf der eine flexible Anbindung vieler räumlich verteilter Geräteteilnehmer über einen Vernetzungsstrang zu realisieren

ist [Schrom 2007]. Diverse physikalisch bedingte Faktoren nehmen damit Einfluss auf das Gesamtverhalten des Kommunikationssystems, wodurch folglich eine signifikante Degradation der Signalqualität auf den Datenleitungen resultieren kann. In der Gebäudeautomation ergeben sich zusätzlich, bedingt durch die individuelle Gebäudearchitektur und der frei wählbaren Installationsorte der Geräte, entsprechend viele Varianten an räumlichen Ausprägungen und Auslegungen physikalischer Topologien. Eine Analyse der Signalqualität (Abschnitt 4.1) ist daher für jede Anwendungsimplementierung separat zu behandeln.
Insbesondere die Verwendung eines standardisierten und ganzheitlichen Datenübertragungssystems setzt eine einwandfreie Funktionsfähigkeit und damit eine hinreichend genaue Untersuchung seiner nichtfunktionalen Eigenschaften, z. B. die Signalqualität auf den Datenleitungen, voraus. Aktuelle Forschungsarbeiten dazu beziehen sich im Wesentlichen auf standardisierte Systeme und legen den Fokus vorwiegend auf automobile Anwendungen [Gunther et al. 2010], [Krammer et al. 2010]. Messtechnische und simulative Untersuchungen sind bisher ausschließlich an Beispielen kleinerer Schaltungsaufbauten durchgeführt worden, da im Vergleich zu GA-Systemen in Fahrzeugen lediglich bis zu 100 Teilnehmer in überschaubarer räumlicher Distanz miteinander vernetzt sind. In [Gunther et al. 2010] wurden Methoden zur Parametersuche und Optimierung der Topologie eines Kommunikationssystems aus dem Bereich der Fahrzeugtechnik vorgestellt. In [Kraft 2002], [Krammer et al. 2010], [Jinrui Nan/Wu Men 2008] und [Muller/Valle 2010] sind Simulationen von FlexRay- und CAN-Bussen unter Variation physikalischer Parameter behandelt worden. In [Prodanov et al. 2009] wurde untersucht, welchen Einfluss verschiedene Konfigurationen eines CAN-Transceivers auf die Signalqualität haben. Die Arbeiten in [Nguyen et al. 2008], [Zdenek/Jiri 2013] und [Wang et al. 2012] legten den Fokus der Parametervariationen auf den Einfluss von EMV-Komponenten innerhalb eines CAN vernetzten Systems.

Verteilte Automatisierungssysteme nutzen sowohl *gemeinsame* als auch *verteilte Ressourcen*, die jeweils weitere Herausforderungen mit sich bringen. Zum einen verwenden beispielsweise Feldbussysteme einen gemeinsamen Übertragungskanal, wodurch es zu eingeschränkten Effizienzeigenschaften aufgrund von auslastungsbedingten Verzögerungszeiten kommen kann (Abschnitt 5.1). Zum anderen sind bei der Verknüpfung verteilter Anwendungsprogramme Zyklusübergänge zu berücksichtigen, die ebenfalls Ursache von Verzögerungen sind (Abschnitt 5.2). Im Anwendungsgebiet der Gebäudeautomation ist das Antwortzeitverhalten bei einer unmittelbar erwarteten Reaktion auf einen Nutzereingriff oder bei der Realisierung sicherheitsrelevanter Systeme, beispielsweise bei Rauchmeldeanlagen relevant.

Weiterhin sind genaue Zeitinformationen notwendig, um Fehlverhalten analysieren zu können, beispielsweise bei der Erkennung schleichender Zustandsverschlechterungen während des Betriebes [Neumann et al. 2011].
In [Greifeneder et al. 2007] werden geeignete Konzepte für die Analyse zeitabhängiger Eigenschaften von Automatisierungssystemen vorgestellt. Für die dort aufgeführten simulativen Verfahren steht eine Fülle von Werkzeugen zur Verfügung (CPNtools, TimeNet, π-Tool, TrueTime, Dymola), die ereignisdiskrete aber auch kontinuierliche Modellierungsansätze erlauben. Weiterhin gibt es bereits einige Anwendungen für die Bewertung des Zeitverhaltens von verteilten Automatisierungssystemen. In dem Projekt *Ekreit* wurden ausgedehnte, dezentrale Systeme mit kleinen Recheneinheiten hinsichtlich ihrer Echtzeitfähigkeit umfangreich analysiert, um ein korrektes Zeitverhalten nachzuweisen und Systemoptimierungen durchführen zu können [Schrom et al. 2012]. Mittels der Simulation von Petrinetzmodellen erfolgten in [Marsal 2008] oder [Singh et al. 2012] Untersuchungen netzwerkbasierter Systeme und in [Ding et al. 2009] und [Liu et al. 2013] dynamische Analysen von CAN-Netzwerken.

Die Möglichkeit nebenläufige Anwendungen in einem verteilten Automatisierungssystem verknüpfen zu können kann zu einem signifikant steigenden *Funktionsumfang* führen, der hinsichtlich der Gewährleistung der Funktionalität schwer überschaubar wird. Im Bereich der Gebäudeautomation betrifft dies u.a. gewerkeübergreifende Funktionen, wie beispielsweise den Datenaustausch zwischen einer Raumregelung und einem Raumbuchungssystem. Weiterhin erfordern neue funktionale Anforderungen wie beispielsweise sicherheitsrelevante AAL-Anwendungen die Einbindung neuer Funktionen. Mit steigender Funktionalität entsteht ein zunehmend komplexeres Abhängigkeitsnetz diverser dezentralisierter Anwendungsfunktionen. Daher ist ein GA-System bei der Inbetriebnahme auf eine korrekte Funktionalität hin zu testen (Abschnitt 7.2.1). Für die Generierung und Erstellung von Testfällen für dezentrale Automatisierungssysteme sind u. a. in [Horstmann 2005] oder [Krause 2012] Methoden vorgestellt worden, die auf Petrinetzen basieren. Eine Anwendung von Testmethoden im Anwendungsfeld der Gebäudeautomatisierung ist u. a. in [Graditi et al. 2013] beschrieben.

Mit steigender Funktionalität steigt auch die Anzahl *partieller Fehler* innerhalb eines verteilten Automatisierungssystems. Deren Ursachen sind neben einer fehlerhaften Programmierung, Parametrierung und Verstellung [Déqué et al. 2000] in physikalisch bedingten Hardwareausfällen, Kommunikationsfehlern oder Änderungen der Regelstrecke zu suchen. Die Auswirkung eines partiellen Fehlers kann eine signifikante Abweichung vom spezifizierten Betriebsverhalten sein, wodurch beispielsweise die

Anforderung einer energieeffizienten Betriebsführung eines Gebäudes nicht mehr erfüllt wird [Plesser et al. 2012]. Die kontinuierliche Betriebsüberwachung eines Gebäudeautomatisierungssystems ist daher ebenfalls zu gewährleisten und wird u. a. in [Sallans et al. 2006], [Saki et al. 2011], [Gomes et al. 2007] und [Diekhake/Schnieder 2013], bzw. in Abschnitt 7.2.2 der vorliegenden Arbeit thematisiert.

Insbesondere Softwarefehler stellen in einem verteilten Automatisierungssystem auch ein Risiko für die *Sicherheit* (*Security* und *Safety*) dar. Der erforderliche Austausch zwischen Gerätekomponenten geht bei potentiell vorhandenen Sicherheitslücken und insbesondere bei funkbasierten Verbindungen [Granzer et al. 2010a] mit der Möglichkeit einer missbräuchlichen Verwendung von Informationen einher. Aus dem Anwendungsfeld der Gebäudeautomatisierung sei beispielsweise die Analyse des typischen Verhaltensprofils einer Person in Form eines passiven Angriffs oder die Kontrollübernahme einer Ressource, beispielsweise eines Aktors, als ein aktiver Angriff zu nennen. [Granzer et al. 2010b] stellte fest, dass vorhandene Standards und Technologien der Gebäudeautomatisierung unterschiedliche und weitgehend unzureichende Schutzmaßnahmen vor Sicherheitsangriffen aufweisen. In einem kooperativen Verbund von Systemen mit verschiedenen ausgeprägten Schutzmechanismen und einer Anbindung über IT- und internetgestützte Technologien, die die Angriffsfläche deutlich vergrößern [Pohlmann 2014], [Störtkuhl 2014], wächst das Sicherheitsrisiko für das gesamte GA-System weiter an. Im Bereich Safety nehmen die Relevanz und der Aufwand einer Gefährdungs- und Risikoanalyse als eine präventive Sicherheitsmaßnahme u. a. aufgrund der steigenden Einbindung sicherheitsrelevanter Funktionen (z. B. AAL-Funktionen) weiter zu. In [Dongbo et al. 2008] und [Kastner/Novak 2009] sind daher bereits Sicherheitsstandards vorgestellt worden, die für die Gebäudeautomatisierung als relevant erachtet wurden.

Die oben dargelegten Problemstellungen sowie die sich ergebenen Anforderungen für verteilte Automatisierungssysteme werden in Tabelle 3 sind zusammengefasst und beispielhaft für das Anwendungsgebiet der Gebäudeautomatisierung konkretisiert. Weiterhin sind in der rechten Spalte mögliche Lösungsansätze, Maßnahmen bzw. Analyseverfahren dargestellt.

Tabelle 3: Darstellung von Problemen, Anforderungen und Lösungsansätzen verteilter (Gebäude-)Automatisierungssysteme

Allgemeine Problemstellung verteilter Systeme	***Anforderungen***	***Spezifische Problemstellungen / Anforderungen aus dem Anwendungsfeld der Gebäudeautomatisierung (Beispiele)***	***Lösungsansätze / Maßnahmen / Analyseverfahren (Auswahl)***
Komplexität	Gerätevernetzung	Vielfalt und Vielzahl von Systemen und Gerätekomponenten	Reglementierende / Offene Interoperabilität
	Systemverständnis		Wissens-repräsentation
Einflussnehmende Faktoren	Funktionsfähigkeit	Signalqualität auf dem Feldbus, EMV	Simulation, Messung
Gemeinsame Ressource	Effizienz	Buslastverhalten	Simulation, Messung
Verteilte Ressourcen		Zyklusübergangsverhalten	
Parallelität	Beobachtbarkeit/ Steuerbarkeit	Zuordnung von Ereignissen	Strukturelle Analyse
Funktionsumfang	Funktionalität	Gewerkeübergreifende Automatisierung	Funktionstest
Partielle Fehler	Stabilität	Reglerverstellungen	Monitoring
Sicherheit	Safety und Security	Sicherheitsrelevante Anwendung, Kontrollübernahme eines Aktors	Gefährdungs- und Risikoanalyse, Monitoring

1.3 Analysen im Lebenszyklus eines Automatisierungssystems

Aus den in Abschnitt 1.2 erläuterten vielfältigen Problemstellungen verteilter Automatisierungssysteme wird ersichtlich, dass in allen Lebenszyklusphasen eines verteilten Systems vielfältige Analysetätigkeiten notwendig sind. Dem entsprechend steht eine Vielfalt an Analyseverfahren und -lösungen zur Verfügung. Diese Vielfalt an Beschreibungsmittel, Methoden und Werkzeuge (BMW) wird in diesem Abschnitt verdeutlicht. Dazu werden im Folgenden typische Analysetätigkeiten aus den einzelnen Phasen des Lebenszyklus eines Automatisierungssystems (Abbildung 1.3) näher erörtert und häufig verwendete Beschreibungsmittel angemerkt.

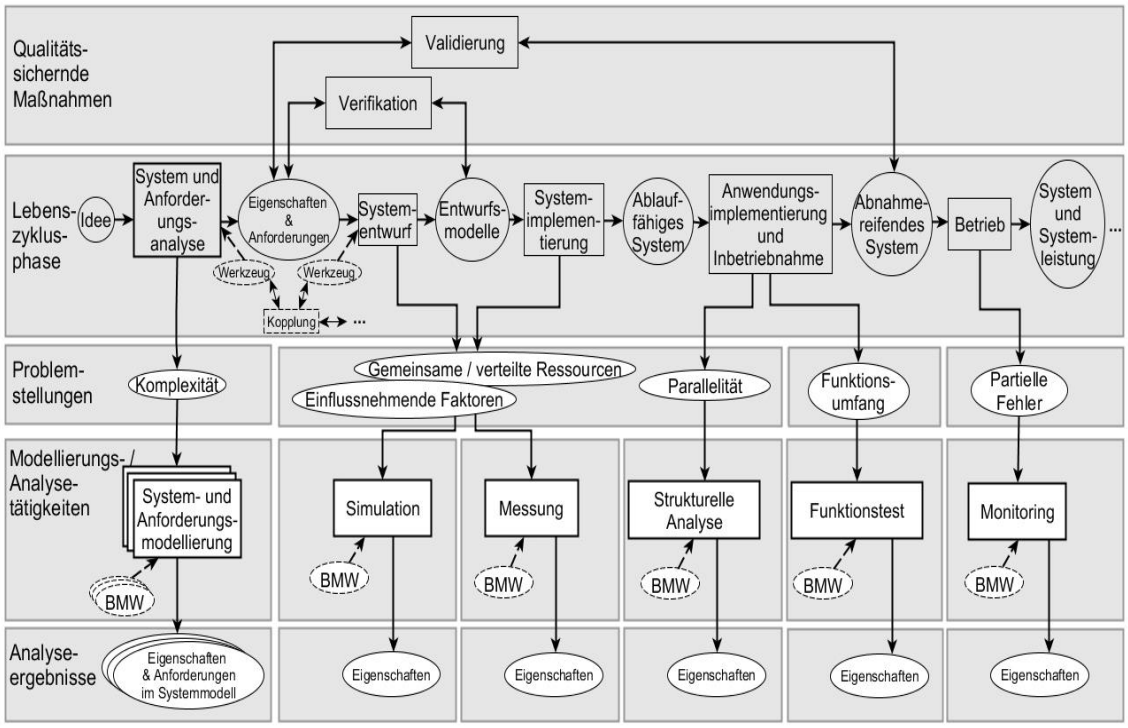


Abbildung 1.3: Vorherrschende Analysetätigkeiten im Lebenszyklus eines Automatisierungssystems

Um von Beginn des Entwicklungsprozesses an die Komplexität beherrschbar zu halten und das allgemeine Systemverständnis zu erhöhen, bedarf es zunächst einer geeigneten *System- und Anforderungsbeschreibung*. Die System- und Anforderungsbeschreibung kann als Situationsanalyse aufgefasst werden, d. h. sie dient, bezogen auf die jeweilige Problemstellung, der Festlegung der Aufgabenstellung, der Klärung der Ausgangssituation und der Steigerung eines gemeinsamen Systemverständnisses. Damit werden mehrere Zielsetzungen verfolgt: zum einen müssen die Anforderungen sowie Eigenschaften des Systems erhoben und klassifiziert werden. Hier können verschiedene natürlich-sprachliche (z. B.: Prosatext, Lastenheft nach [DIN 69901-1], Satzschablonen)

oder modellbasierte Beschreibungsmittel (z. B.: Klassendiagramme, Funktions- und Verhaltensmodelle) zum Einsatz kommen. Zum anderen kann in dieser Phase eine Begriffsanalyse erfolgen, die der Vereinheitlichung der Sprache, beispielsweise durch die Sammlung und Definition von (Fach-)Begriffen dient. Neben der Bestimmung allgemeiner und fachspezifischer Begriffe durch deren Definitionen über ein Glossar umfasst eine weiterführende terminologische Analyse auch die Angabe von Merkmalsausprägungen eines Begriffes durch deren Attribuierung [Schnieder/Schnieder 2012]. Weiterhin lassen sich die Begriffe nach gewissen Kriterien (z. B. Varietät, Abstraktionsniveau, Sprachkategorie, usw.) klassifizieren [Hesse et al. 1994] und die Beziehungen zwischen den Begriffen durch Relationen beschreiben [Schnieder/Schnieder 2012]. Folglich lässt sich ein Begriffsystem modellieren, welches vorzugsweise über Klassendiagramme beschrieben wird.

In der Phase des *Systementwurfs* sind Synthese-Analysen (z. B. Simulationen) u. a. zur frühzeitigen Bewertung von nichtfunktionalen Qualitätsmerkmalen erforderlich, um so teure Korrekturen innerhalb ungewollter Iterationen in den späteren Phasen der Errichtung verhindert zu können. Sofern eine Systemspezifikation als formales Strukturmodell vorliegt, können durch strukturelle Ansätze der Analyse (z. B. durch eine Erreichbarkeitsanalyse) spezifische Netzeigenschaften wie Lebendigkeit und Beschränktheit analysiert werden. Durch die Simulation eines Analysemodells lassen sich Schwachstellen aufdecken, um ungeeignete Lösungsvarianten schon im anfänglichen Entwicklungsprozess verwerfen zu können, beispielsweise durch Zuverlässigkeitssimulationen. Weiterhin können Simulationen für Sensitivitätsanalysen genutzt werden, indem die Modelle mit variablen Parameterwerten ausgeführt werden. Dadurch lassen sich Lösungsvorschläge und Systemvarianten, wie beispielsweise eine gewählte Systemarchitektur, experimentell evaluieren und optimieren. Simulationen eignen sich außerdem für die Verhaltensbestimmung von Systemen, beispielsweise mittels einer Transientenanalyse von elektrischen Schaltungen oder durch eine Performance-Analyse eines Kommunikationssystems.

Nach bzw. während der *Systemimplementierung* sind durch Messungen am Realsystem weitere nichtfunktionale Eigenschaften zu ermitteln. Aufgrund der Einschränkungen, die ein Modell als Vereinfachung des Realsystems mit sich bringt, ist dieses zu validieren, indem die Simulationsergebnisse mit verlässlichen Resultaten aus Messungen verglichen werden.
Die *anwenderorientierte Implementierung* und Konfiguration eines Automatisierungssystems erfordert weitere Analysen (z. B. eine strukturelle Analyse) und macht eine abschließende Prüfung funktionaler Eigenschaften nach der

Inbetriebnahme des Systems erforderlich (Funktionstest). Für die Analyse nach der anwenderorientierte Implementierung und während der Inbetriebnahme eignen sich u. a. dynamische Testverfahren. Neben den strukturorientierten Tests (White-Box-Tests), die auf Kriterien der Pfadabdeckung konkreter Implementierungen basieren, kommen funktionsorientierte Verfahren (Black-Box-Tests), die gegen die vorgegebene Spezifikation testen, zum Einsatz. Sofern die Spezifikation in Form eines Zustandsautomaten vorliegt (vgl. Abbildung 1.2), eignen sich für die Analyse u. a. auch zustandsbasierte Testmethoden, mit denen das Durchlaufen und Erreichen von Zuständen und Zustandsübergängen geprüft wird. Für die Testbeschreibung bzw. Testausführung werden häufig graphische oder textuelle Beschreibungssprachen wie UTP oder TTCN-3 verwendet.

Letztlich müssen aufgrund der Gefahr von partiellen Fehlern und Systemangriffen sowohl die funktionalen als auch die nicht-funktionalen Eigenschaften während der *Betriebsphase* überwacht werden. Diese Aufgabe können Monitoring-Systeme erfüllen. Die Erfassung von Systemeigenschaften während des Betriebes kann sowohl ereignisgesteuert durch die Triggerung des Monitors aber auch zeitgesteuert durch eine periodische Abfrage (Polling) ermöglicht werden. Abschließend erfolgen im Monitoringprozess die Interpretation der Daten, die Analyse der Zustandsänderungen und die Visualisierung des Systemzustandes. Erfolgt die abschließende Verarbeitung der Monitordaten während des Prozessablaufes spricht man von einem Online-Monitor. Für die verschiedenen Aufgaben eines Monitors kommen verschiedene Beschreibungssprachen zum Einsatz, beispielsweise die Ereignisspur-beschreibungssprache TDL zur Verarbeitung oder Kurvendiagramme zur Präsentation von Monitoringinformationen [Kurschl 2000: 33].

In Abbildung 1.3 ist aufgezeigt, wie die einzelnen Entwicklungsschritte innerhalb des Lebenszyklus eines Systems aufgrund ihres zeitlichen Aufeinanderfolgens aber auch durch qualitätssichernde Maßnahmen wie die *Verifikation* und die *Validierung* horizontal miteinander gekoppelt sind [Meyer zu Hörste 2004: 14 f]. Mithilfe von *Werkzeugketten* können diese Kopplungen weiter verstärkt werden und so zu einer erhöhten Durchgängigkeit des Gesamtprozesses verhelfen.

Im Gegensatz dazu erfolgen die einzelnen Analysetätigkeiten zu den in Abschnitt 1.2 behandelten Problemstellungen isoliert voneinander. Die verwendeten Beschreibungsmittel, Methoden und Werkzeuge sind häufig sehr spezifisch ausgewählt und kommen ausschließlich im individuellen Kontext der jeweiligen Analyseaufgabe zum Einsatz. Eine Wiederverwendung von Betriebsmitteln sowie aus den

Analyseprozessen gewonnenen Modelle und Ergebnisse ist häufig nur eingeschränkt möglich bzw. wird bisher gar nicht in Betracht gezogen.

1.4 Zielsetzung

Wie die vorherigen Abschnitte zeigen, sind für die dargelegten Probleme verteilter Automatisierungssysteme zwar bereits konkrete Analyseansätze vorhanden, deren Vorbereitungen und Durchführungen erfolgen bislang jedoch isoliert voneinander, d. h. es werden keine verknüpfenden Zusammenhänge zu vorangegangenen bzw. nachfolgenden Analyseansätzen hergestellt. Aus diesem Umstand heraus entwickelte sich eine große Vielfalt an spezifischen Beschreibungsmitteln, Methoden und Werkzeugen, deren individueller Einsatz einen insgesamt erhöhten Analyseaufwand nach sich führt.

Die vorliegende Arbeit behandelt daher die Analyse verteilter Systeme unter der Anwendung eines vereinheitlichten und systematischen Vorgehens. Unter dem Einsatz weniger, geeigneter Beschreibungsmittel, probaten Methoden und vielseitig einsetzbaren Werkzeuge wird eine strukturierte Modellierung, Simulation und Überwachung eines verteilten Automatisierungssystems demonstriert. Die Anwendung des Vorgehens zur Ermittlung (nicht-)funktionale Eigenschaften erfolgt am Beispiel eines verteilten Gebäudeautomatisierungssystems.

Das erste Teilziel beinhaltet die Konzepterstellung für eine systematische Analyse auf Basis des BMW-Prinzips [Schnieder 1999: 18]. Es werden geeignete Beschreibungsmittel, Methoden und Werkzeuge ausgewählt, die den Anforderungen an Einfachheit, Wiederverwendbarkeit und Integrierbarkeit gerecht werden. Dadurch kann eine zusammenführende Wirkung zwischen den einzelnen Analysetätigkeiten herbeigeführt und ein phasenübergreifendes vereinheitlichtes Analysekonzept geschaffen werden.

Das zweite Teilziel ist die Bereitstellung von beschreibenden Systemmodellen zur Wissensrepräsentation und zur Verdeutlichung einer konkreten Analyseaufgabe. Durch Modellierungsansätze mittels eines variablen Abstraktionsgrades und mittels einer eindeutigen Begriffsformalisierung schafft das Systemmodell ein verbessertes und eindeutiges Verständnis für das jeweilige zu analysierende (Teil-)System. Der variable Abstraktionsgrad wird durch die Einführung von Hierarchieebenen erreicht, die sowohl abstrakte sowie konkrete Sichtweisen auf ein System ermöglichen. Dadurch lässt sich ein zu analysierendes System beginnend vom allgemeinen Systembegriff bis hin zur Formulierung konkreter Bauelemente darstellen. Die Anforderung an Verständlichkeit

wird durch eine eindeutige Hierarchisierung von Begriffsattributen erfüllt. Als Ergebnis steht für die jeweilige zu behandelnde Problemstellung eine umfassende und formalisierte Beschreibung des Systems zur Verfügung.

Die Anwendung des Analysekonzeptes für die Ermittlung konkreter nichtfunktionaler Eigenschaften, Merkmale und Kenngrößen eines verteilten Systems, bezogen auf eine konkrete Problemstellung, stellt das dritte Teilziel der Arbeit dar. Für das dieser Arbeit herangezogene Automatisierungssystem aus [Schrom 2007] werden durchgeführte Analysen sowohl für eine Bewertung der Signalqualität auf Datenleitungen als auch für eine Ermittlung effizienzbezogener Eigenschaften vorgestellt.

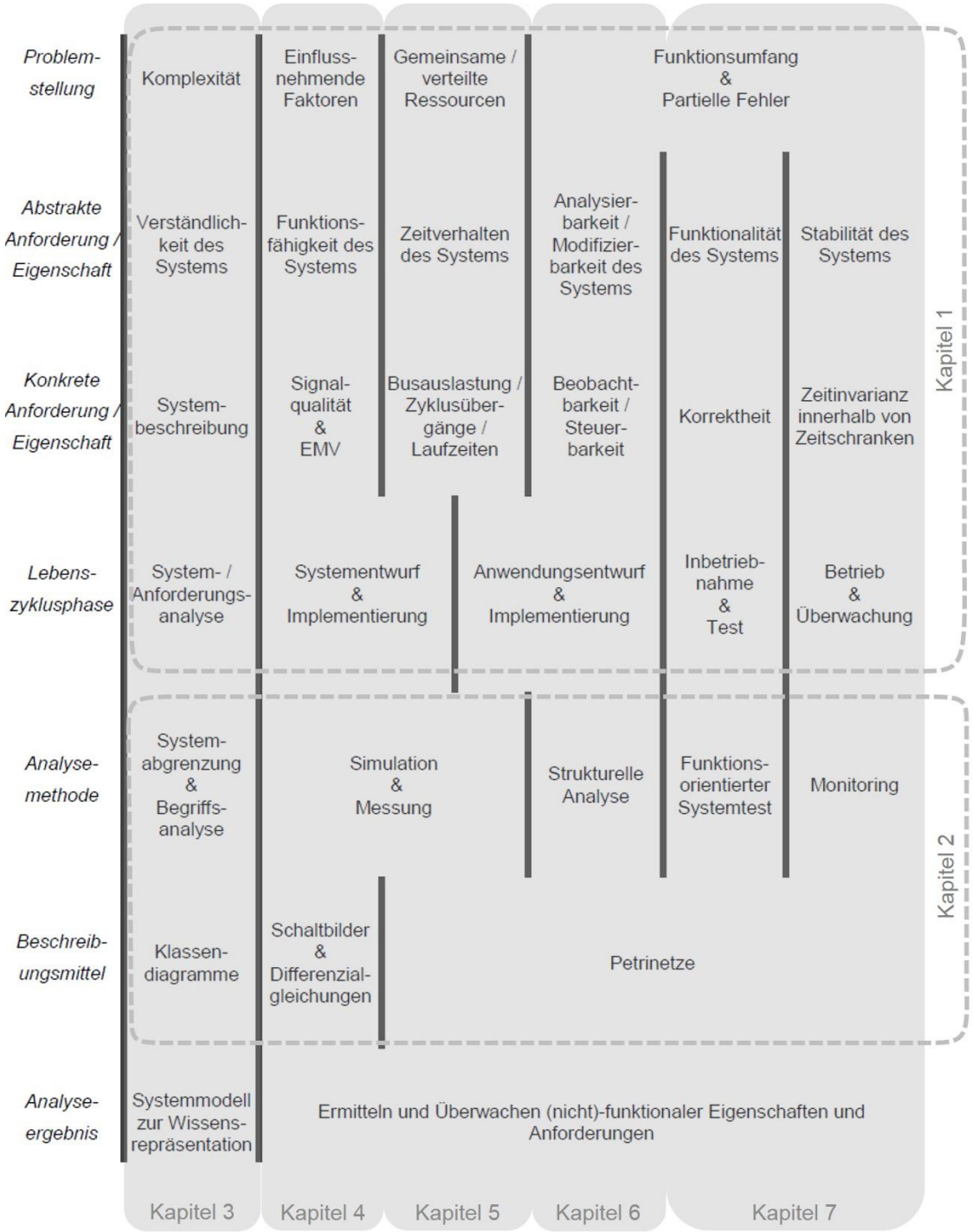
Die Analyse der Signalqualität ist dahingehend von Bedeutung, dass sich der physikalische Aufbau des betrachteten Automatisierungssystems aufgrund seiner abweichenden Konzeptionierung zu anderen standardisierten Realisierungen deutlich von diesen unterscheidet und sich daher mit diesen nicht vergleichen lässt. Dabei liegen die Besonderheit des zu analysierenden Bitübertragungssystems und damit das Hauptaugenmerk der Untersuchung auf einem räumlich weit ausgeprägten Vernetzungsstrang mit einer großen Anzahl angeschlossener Kommunikations-teilnehmer.
Zeitliche Restriktionen stellen ebenso eine spezifische Randbedingung in verteilten Automatisierungssystemen dar. Als weitere nichtfunktionale Eigenschaft wird daher das temporale Verhalten behandelt, welches u. a. von der Aufteilung der einzelnen Prozesse auf die dezentral angeordneten Programm- und Gerätekomponenten abhängt, d. h. von dem sich ergebenen Datenübertragungs- und Programmübergangsverhalten bestimmt wird.

Das vierte Teilziel dieser Arbeit sieht eine Online-Analyse des behandelten Automatisierungssystems in den Phasen der Inbetriebnahme und des operativen Betriebes vor. Als Anwendungsbeispiele werden die Ausführung eines Testfalles sowie eine Online-Laufzeitprüfung behandelt. Zur Schaffung der Voraussetzung für die Durchführung einer Online-Analyse sind zunächst die Eigenschaften der Beobachtbarkeit und Steuerbarkeit des zu analysierenden (Teil-)Systems zu bewerten. Weiterhin wird eine Analyse-Umgebung benötigt, die die Modelle für die Testfallausführung sowie für das Online-Monitoring von Laufzeiten bereitstellt und diese Modelle an das Realsystems koppelt.

1.5 Aufbau der Arbeit

Aus der im dem vorherigen Abschnitt erläuterten Zielsetzung leitet sich unmittelbar die Struktur dieser Arbeit ab, die in Tabelle 4 zusammenfassend dargestellt ist. In Kapitel 2 wird auf das Konzept zur systematischen und vereinheitlichten Analyse eines verteilten Automatisierungssystems eingegangen und die in dieser Arbeit verwendeten Beschreibungsmittel, Methoden und Werkzeuge erörtert. In Kapitel 3 wird die Vorgehensweise zur Erstellung wissensrepräsentierender Systemmodelle, die die Grundlage des Analysekonzeptes darstellen, mit Hilfe von Beispielen vorgestellt. Auf Basis der klassischen Strukturanalyse und eines neuartigen terminologischen Analyseansatzes entsteht so eine detaillierte Beschreibung des zu analysierenden Systems. Neben der methodischen Beschreibung des vorhandenen Systems wird in dem Kapitel weiterhin eine Anforderungserhebung eingeführt, um auch die zu erfüllenden Eigenschaften abzubilden, die für die jeweilige Problemstellung relevant sind. Bezugnehmend auf die zu ermittelnden nichtfunktionalen Eigenschaften werden in Kapitel 4 und Kapitel 5 die durchgeführten Verhaltens- und Sensitivitätsanalysen vorgestellt. Kapitel 4 behandelt die Analysetätigkeiten hinsichtlich der Signalqualität und der elektromagnetischen Verträglichkeit. In Kapitel 5 wird auf das Zeitverhalten verteilter Automatisierungssysteme eingegangen. Die vorgestellten Untersuchungen beziehen sich dabei sowohl auf systemspezifische Implementierungen als auch auf das in Abschnitt 1.1.1 eingeführte konkrete Anwendungsbeispiel. In Kapitel 6 werden die Systemeigenschaften der Beobachtbarkeit und der Steuerbarkeit als wesentliche Voraussetzung für die Durchführung von Online-Analysen behandelt. Die Umsetzung einer Online-Analyseumgebung mittels Petrinetzen und die auf Basis der Beobachtbarkeits- und der Steuerbarkeitsanalyse entwickelten Modelle zur Testfallausführung und Laufzeitüberwachung werden in Kapitel 7 vorgestellt. Die Ergebnisse der Arbeit werden in Kapitel 8 zusammengefasst. Zudem gibt das Kapitel einen Ausblick auf zukünftige Arbeiten im Kontext des vorgestellten Analysekonzeptes.

Tabelle 4: Gliederung und Fokus dieser Arbeit



2 Ziel und Konzept einer vereinheitlichten Modellierung und Analyse

Im einleitenden Kapitel 1 wurde das breite Spektrum an Anwendungsfeldern und Problemen von verteilten Automatisierungssystemen aufgezeigt. Den vielen Problemstellungen aus unterschiedlichen Anwendungsfeldern stehen in den verschiedenen Lebenszyklusphasen wiederum zahlreiche Modellierungs- und Analysemöglichkeiten gegenüber. Entsprechend existiert eine große Vielfalt an Beschreibungsmitteln, Methoden und Werkzeuge (BMW), die zumeist spezifisch sind und lediglich isoliert voneinander zur Anwendung kommen. Dadurch steigt der Gesamtaufwand für die Modellierung und Analyse insbesondere für verteilte Systeme deutlich an. Um eine Verknüpfung der einzelnen Analyseaufgaben und dadurch eine Reduzierung des Aufwandes zu bewirken, bedarf es eines systematischen und vereinheitlichten Konzeptes, welches im folgenden Kapitel vorgestellt wird. Durch eine strategische Auswahl von Beschreibungsmitteln, Methoden und Werkzeuge können die einzelnen Modellierungs- und Analysetätigkeiten aufeinander aufbauend gestaltet bzw. zusammengefügt werden, beispielsweise in Form einer Wiederverwendbarkeit bzw. einer Integration von Beschreibungsmitteln, Methoden und Werkzeuge sowie Modellen und vorangegangenen Analyseergebnissen.

2.1 Anforderungen

Um zur Vereinheitlichung der Modellierung und der Analyse eine geeignete Auswahl an Beschreibungsmittel, Methoden und Werkzeugen zu treffen und durch deren Verknüpfungen Zusammenhänge zwischen den einzelnen Modellierungs- und Analysetätigkeiten zu schaffen, sind folgende Anforderungen an das Analysekonzept zu stellen:

Beschreibungsmittel, Methoden und Werkzeuge

- Verwendung modularer und hierarchischer Modellierungsansätze
- Verwendung geeigneter Modellkonzepte und allgemein nutzbarer Beschreibungsmittel, Methoden und Werkzeuge
- Verwendung phasenübergreifender Beschreibungsmittel und Werkzeuge
- Verwendung bereits vorhandener Methoden
- Rechnergestützte Modellierung mit Hilfe geeigneter Werkzeuge
- Automatisierte Modellgenerierung.

Integration

- Erstellung einer einheitlichen Basis zur Wissensrepräsentation
- Schaffung von Synergien zwischen Modellen (modellzentrierte Integration)
- Kopplung von Werkzeugen mittels Übergabestellen (kooperativer Werkzeugverbund)
- Integration des Realsystems in das Analysekonzept
- Zusammenfassende Darstellung der Analyseergebnisse.

Aus den genannten Anforderungen leitet sich das Analyse-Konstrukt aus Abbildung 2.1 ab, welches in den Lebenszyklus des Automatisierungssystems gemäß Abbildung 1.3 eingebettet ist.

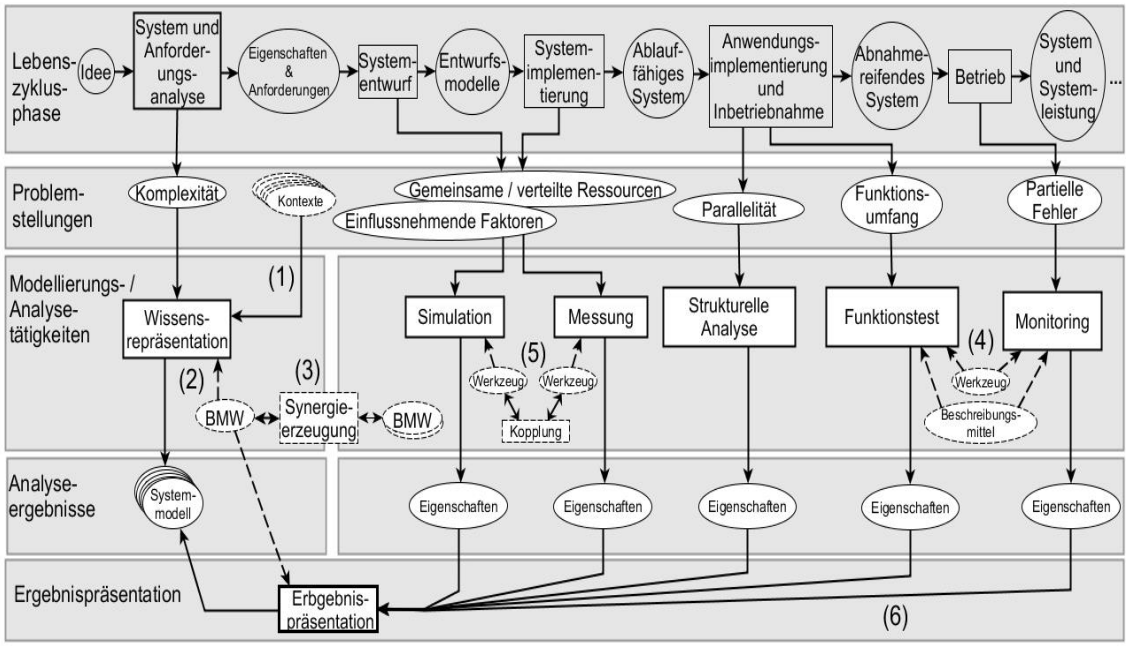


Abbildung 2.1: Darstellung des Lebenszyklus eines Automatisierungssystems und exemplarisch skizzierte Ansätze zur Vereinheitlichung der Analysetätigkeiten

Die Darstellung stellt exemplarisch folgende horizontale Verknüpfungen zwischen den Problemstellungen bzw. den Analysetätigkeiten und -ergebnissen dar:

(1) Beschreibung des Kontextes [Schnieder 1999: 18] einer spezifischen Analyseaufgabe auf Basis einer einheitlichen Wissensrepräsentation

(2) Verwendung eines Beschreibungsmittels, einer Methode und eines Werkzeuges für die Ergebnispräsentation und die Wissensrepräsentation

(3) Synergieerzeugung zwischen Beschreibungsmitteln zwecks des Informationsaustausches zwischen statischen Modellen zur Wissensrepräsentation und dynamischen ausführbaren Analysemodellen

(4) Verwendung eines Beschreibungsmittels und eines Werkzeuges für mehrere Analysetätigkeiten

(5) Kooperativer Werkzeugverbund

(6) Zusammenfassende Ergebnispräsentation.

2.2 Konzept und Methoden

Im folgenden Abschnitt wird das aus den Anforderungen abgeleitete Konzept, welches in Abbildung 2.2 skizziert ist, im Detail vorgestellt. In Anlehnung an die im vorherigen Abschnitt aufgestellten Kriterien wird auf das Vorgehen, die verwendeten Methoden, die Beschreibungsmittel und die Werkzeuge als auch auf die Integrationsansätze dieser Betriebsmittel näher eingegangen.

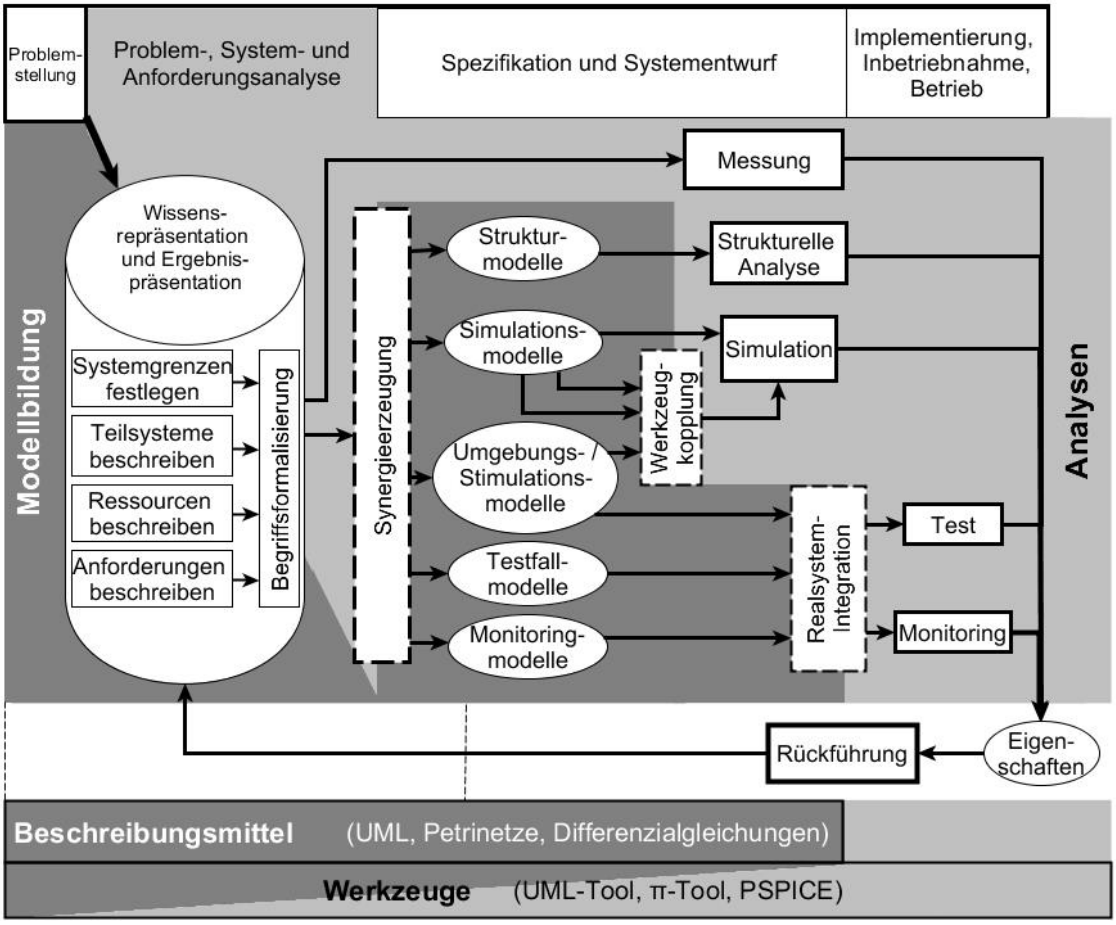


Abbildung 2.2: Konzept zur vereinheitlichten Analyse eines Automatisierungssystems

Ausgangspunkt des Vorgehens ist eine konkrete *Problemstellung* bzw. Anforderung (Signalqualität, Busauslastung, Beobachtbarkeit, usw.) aus der sich ein konkreter Analysebedarf ergibt (vgl. Abschnitt 1.2).

Zur *Wissensrepräsentation* von Systeminformationen und zur *Ergebnispräsentation* von Analyseergebnissen wird ein auf die Problemstellung zugeschnittenes Begriffssystem formuliert. Das schrittweise Erstellen dieses Systemmodells wird in Kapitel 3 im Detail und unter Verwendung von Beispielen näher erörtert. Das Modell beinhaltet u. a. die Ergebnisse aus den grundlegenden Prozessschritten der *klassischen Systemanalyse*. Dazu gehören die eindeutige Unterscheidungsfindung des relevanten (Teil-)Systems zu seiner Umwelt, das Feststellen der Systemelemente und deren Beziehungen, die systematische Erhebung und Quantifizierung der Systemeigenschaften sowie der auf die Problemstellung bezogenen Anforderungen. Für die Darstellung des Systemmodells eignen sich zum Beispiel Klassendiagramme mit UML/SysML-Notation, deren Modellierung durch rechnergestützte Werkzeuge wie Argo UML, Dia oder Umbrello erfolgen können. Das Werkzeug *iglos* bietet ergänzend eine Unterstützung zur terminologischen Analyse [Stein et al. 2010: 18]. Durch eine *Begriffsformalisierung* lassen sich die Informationen aus dem statischen Systemmodell formal interpretieren und eindeutig zuordnen. Damit stellt das Systemmodell eine detailliert beschreibende Modellbasis dar.

Im Fall einer anschließenden messtechnischen Analyse dient das wissensbeschreibende Systemmodell der Messvorbereitung, beispielsweise durch die Beschreibung des Messaufbaus.

Das formale Systemmodell kann weiterhin für die Erstellung nachstehender dynamischer Analysemodelle genutzt werden. Um dafür eine *Synergieerzeugung zwischen den Beschreibungsmitteln* des Systemmodells und den nachfolgenden ausführbaren Analysemodellen (*Strukturmodelle, Simulationsmodelle, Umgebungs- bzw. Stimulationsmodelle, Testfall-* und *Monitoringmodelle*) möglichst einfach und einheitlich zu gestalten, ist es wünschenswert für deren Erstellung möglichst wenige phasenübergreifende Beschreibungsmittel zu verwenden (Abschnitt 2.2.1).
Die Modellierung und die anschließende Ausführung der Analysemodelle erfolgt mit Unterstützung geeigneter Werkzeuge, die in Abschnitt 2.2.2 vorgestellt werden. Um verschiedene Modelltypen in einem Verbund ausführen zu können, ist eine *Werkzeugkopplung* zu ermöglichen. Eine Kopplung zwischen einem Petrinetzwerkzeug und einem schaltungstechnischen Simulator wird u. a. in [Donath et al. 1995] empfohlen und in Abschnitt 4.1 der vorliegenden Arbeit behandelt (Co-Simulation).

Für die Durchführung von Systemtests oder zur Überwachung des Systems während der Betriebsphase muss das Automatisierungssystem selbst an die Umgebungs- bzw. Stimulationsmodelle, Testfallmodelle oder Monitoringmodelle angebunden werden. Für verteilte Systeme eignet sich dafür der Kommunikationskanal als Koppelstelle. Über diesen können Telegramme empfangen und den Analysemodellen zur Verfügung gestellt werden. In Gegenrichtung kann durch das Ablegen eines Telegrammes auf den Kommunikationskanal das Realsystem stimuliert werden. Die Realisierung einer solchen Online-Kopplung zur *Integration des Realsystems* an Online-Analysemodelle wird in Abschnitt 7.1 vorgestellt. Gemäß der Anforderung an das vereinheitlichende Analysekonzept, möglichst wenige Beschreibungsmittel anzuwenden, basiert die Koppelstelle ebenfalls auf Petrinetzmodellen.
Das Analysekonzept schließt mit der *Rückführung* der durch die Analysetätigkeiten gewonnenen Ergebnisse in das Systemmodell ab. Durch eine zweckmäßige Einordnung der ermittelten Eigenschaften in das Systemmodell wird dieses somit zunehmend mit Wissen angereichert

2.2.1 Beschreibungsmittel

Eine geeignete Modellierungssprache, welche sowohl die verschiedenen Modellierungskonzepte als auch sämtliche Lebenszyklusphasen abdeckt, sind Petrinetze. Diese weisen folgende wesentliche Vorteile auf:

Einsatzbereich

Der Einsatzbereich von Petrinetzen ist vielfältig. Sie können sowohl für die Spezifikation als auch für implementierungs- und betriebsnahe Anwendungen genutzt werden. Damit stellen Petrinetze ein phasenübergreifendes Beschreibungsmittel dar [Schnieder 1999], [Girault/Valk 2003], [David/Alla 2001].

Theorie:

Die mathematische Beschreibung von Petrinetzen ermöglicht die Ermittlung und Verifikation spezifischer Netzeigenschaften wie beispielsweise der Lebendigkeit und Erreichbarkeit auf Basis graphentheoretischer und algebrahischer Untersuchungen. Die formale Darstellung der Petrinetze stellt eine Basis zur Synergiebildung mit anderen Beschreibungsmitteln dar [Starke 1990].

Einfache Beschreibungssprache:

Petrinetze bestehen aus wenigen Elementen und sind daher leicht erlernbar bzw. eignen sich für eine automatisierte Modellgenerierung [Abel 1990], [Reisig 1982], [Reisig 2013].

Abstraktion:

Durch Verfeinerungs- und Vergröberungskonzepte lassen sich Petrinetze auf unterschiedlichen Abstraktionsebenen modellieren, hierachisch aufbauen und modular gestalten [Jensen 1992].

Simulation:

Durch die Simulation eines Petrinetzmodells mit Zeitattributierung können quantitative Aussagen sowohl über funktionale Eigenschaften (z. B. Zeitverhalten [AimoneMarsan et. al. 1986]) als auch über nichtfunktionale Eigenschaften (z. B. Verfügbarkeit [Schneeweiss 1999]) getroffen werden.

Graphische Darstellung:

Die graphische Repräsentation eines Petrinetzes führt eine verständliche und anschauliche Darstellung herbei. Die graphische Darstellung kann insbesondere in einem Monitorsystem genutzt werden, indem der aktuelle Systemzustand über eine Marke visualisiert wird [Oberquelle 1981].

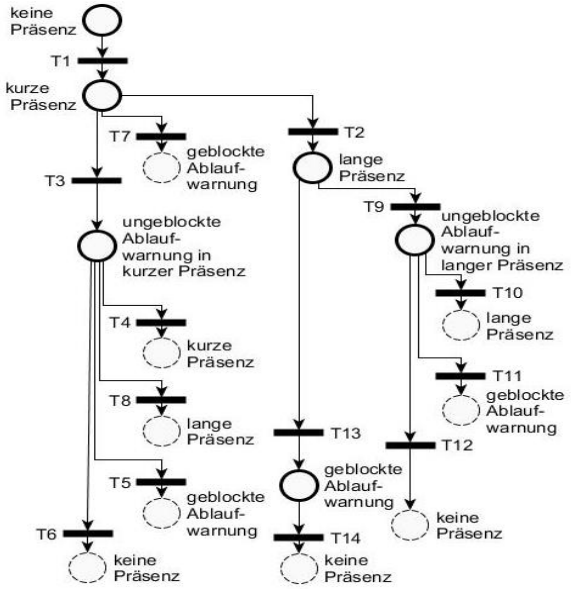
Werkzeugverfügbarkeit:

Es steht eine umfangreiches Angebot an Werkzeugen (vgl. Tabelle 6 in Abschnitt 2.2.2) für eine rechnergestützte Modellierung und Analyse von Petrinetzen zur Verfügung. [Quiroga et al. 2014]

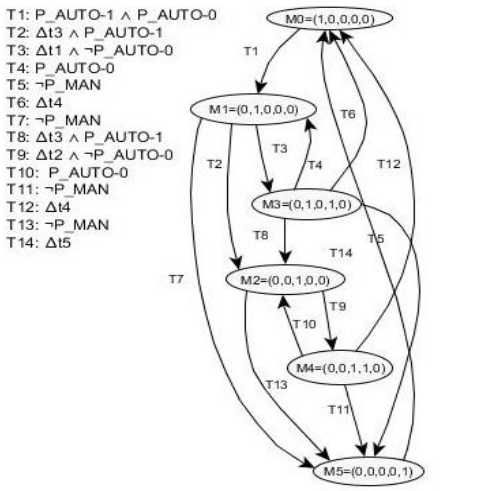
Der vielfältige Einsatz von Petrinetz-Modellen für unterschiedliche Analysetätigkeiten in verschiedenen Lebenszyklusphasen eines Systems wird im Folgenden exemplarisch aufgezeigt. Auf Basis des eingeführten Beispiels zur Belegungsauswertung (Abschnitt 1.1.1) werden in Abbildung 2.3 vier Netze skizziert, die für die *Systembeschreibung*, die *strukturelle Analyse*, die *Simulation* sowie für den *Funktionstest* (Online-Analyse) in den Phasen der *Systemanalyse*, des *Entwurfs* sowie während der *Inbetriebnahme* bzw. im operativen *Betrieb* zur Anwendung kommen können (vgl. Abbildung 2.1).
Exemplarisch für ein *Strukturmodell* ist in Abbildung 2.3 (links oben) die Zustandsmaschine für die Funktion zur Belegungsauswertung dargestellt

(Abbildung 1.2). Solche statischen Modelle dienen im Wesentlichen der Beschreibung eines Systems durch die Abbildung zentraler Systemeigenschaften (vgl. Kapitel 3) oder können in der Entwurfsphase eines Systems für deren Strukturanalyse genutzt werden (vgl. Kapitel 6).

Zustandsgraph /-maschine (statische Analyse)

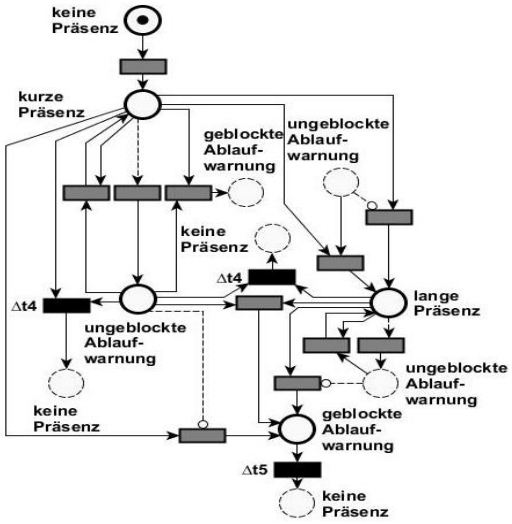


Erreichbarkeitsgraph (dynamische qualitative Analyse)



M=(keine Präsenz, kurze Präsenz, lange Präsenz, ungeblockte Ablaufwarnung, geblockte Ablaufwarnung)

Zeitbewertetes Petrinetz (dynamische quantitative Analyse)



Farbiges Petrinetz und Petrinetz mit externen Stellen (Online-Analyse)

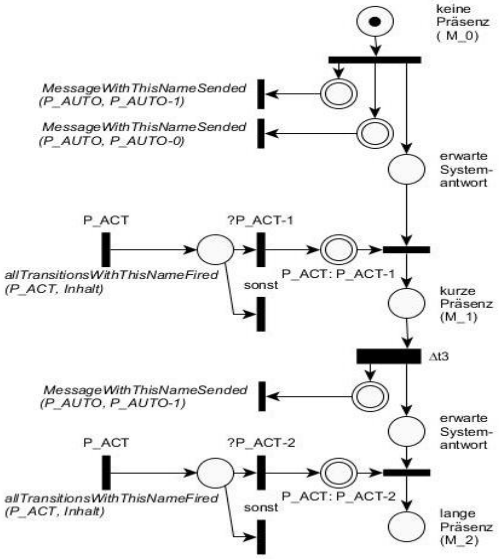


Abbildung 2.3: Netzmodelle für die Analyse am Beispiel der Funktion einer Raumbelegungsauswertung

Für dynamische qualitative Untersuchungen stehen Verfahren wie beispielsweise die Erreichbarkeitsanalyse von Petrinetzen zur Verfügung, mit der eine Verifikation und Prüfung bestimmter Netzeigenschaften wie Lebendigkeit, Reversibilität oder Realisierbarkeit ermöglicht wird. In Abschnitt 6.1 wird beispielsweise auf die Überführung eines Petrinetzes in einen endlichen Automatengraph eingegangen, unter der Voraussetzung, dass der Erreichbarkeitsgraph, aus dem sich die dynamisch qualitative Eigenschaft der Determiniertheit deduzieren lässt, eine endliche Erreichbarkeitsmenge aufweist. In Abbildung 2.3 (rechts oben) ist ein Erreichbarkeitsgraph für die Funktion der Raumbelegungsauswertung dargestellt.

Mit der Einführung zeitbehafteter Netzelemente lassen sich deterministische und stochastische Zeitverbräuche in Form von Zeitintervallen angeben und damit das Portfolio an Analyseverfahren um Simulationen zur Untersuchung dynamisch quantitativer Eigenschaften, beispielsweise Untersuchung der System-Leistung oder für die Zuverlässigkeitsbewertung, erweitern. In Kapitel 5 der vorliegenden Arbeit werden zur Analyse des Zeitverhaltens verteilter Automatisierungssysteme *Simulationsmodelle* vorgestellt, die auf Petrinetzen mit deterministischen und stochastischen Zeitverbräuchen zur Schaltverzögerung aufbauen. In Abbildung 2.3 (links unten) ist exemplarisch ein zeitattribuiertes Petrinetzmodell aufgezeigt.

Liegt eine direkte Kopplung eines realen Steuerungssystems an ein Petrinetzmodell vor, kann dieses zur Steuerung und Beobachtung sowie zur anschließenden Online-Analyse genutzt werden. Der in Abbildung 2.3 (rechts unten) dargestellte Prozess beschreibt die Stimulation eines Realsystems zur Erzwingung von Zustandsübergängen innerhalb der realisierten Funktion (hier: keine Präsenz → kurze Präsenz → lange Präsenz) mit Hilfe von Petrinetzen mit externen Stellen [Hanisch 1992]. Eine anschließende Auswertung des Systemverhaltens erfolgt durch das jeweilige Einlesen von Zustandsinformationen (hier: Präsenzzustand) aus dem Informationsprozess. Auch interpretierte Petrinetze bieten die Möglichkeit der erweiterten Modellierung unter Berücksichtigung von Ein- und Ausgängen [Litz/Frey 1999]. Weiterführende Erläuterungen zu Online-Analysemodellen, die der gezielten Stimulation und dem Auslesen von Zustandsinformationen aus einem Realsystem dienen, erfolgen in Kapitel 7.

Sofern Eingangs- und Ausgangsgrößen der Modelle in binärer Form vorliegen bzw. binär beschreibbar sind, eignet sich eine Modellierung mittels der bis hier eingeführten Netztypen. Da insbesondere Petrinetze vorwiegend Systeme mit wertdiskreten Zustandsübergängen abbilden, erfolgt die Handhabung analoger Größen durch deren binäre Beschreibung. Den wertkontinuierlichen Signaleingängen werden dabei binär beschreibbare Situationen diskret zugeordnet. Ausgangsseitig weisen Analogwert-Umschalter in Abhängigkeit binärer Stellgrößen einem analogen Ausgang einen entsprechenden Festwert zu. Solche binären Beschreibungen analoger Größen sind nicht

mehr zielführend, sobald eine Analyseaufgabe die Berücksichtigung wertkontinuierlicher Signale erforderlich macht. In diesen Fällen sind Ansätze zur analytischen oder numerischen Lösung von Differentialgleichungen (DGL) anzuwenden. Für die Modellierung von Systemen mit zeit- und wertkontinuierlichen Zustandsübergängen stehen zwar u. a. Petrinetze höherer Klassen, beispielsweise kontinuierliche oder hybride Petrinetze zur Verfügung [Bredebusch 1994] [David/Alla 2001], diese stehen aber der Anforderung an der Verwendung einfacher Werkzeuge entgegen [Hummel 2001]. Ebenso ist deren Anwendung für Systeme mit Zeit- und wertekontinuierlichen Zustandsübergängen aufgrund fehlender Modellklassen ungeeignet. In der Praxis haben sich insbesondere für analoge schaltungstechnische Systeme SPICE-basierte Softwarelösungen durchgesetzt (vgl. [Saito 2002], [Guezgouz et al. 2010]).

Die dem Analysekonzept zugrundeliegenden Beschreibungmittel (Klassendiagramme, Petrinetze und Differentialgleichungen) sind bezugnehmend auf die Klassifizierungskriterien nach [Page/Liebert 1991] in Tabelle 5 zusammengefasst. Es wird ersichtlich, dass durch die eingeführten Beschreibungsmittel die unterschiedlichen Modellkonzepte umfassend abgedeckt werden.

Tabelle 5: Modellkonzepte verschiedener Beschreibungsmittel

	Beschreibungmittel		
Klassifizierung	**K/I-Netze / Klassen-Diagramme / Netzlisten**	**Petrinetze mit Marken**	**DGL / SPICE / (Kontinuierliche Petrinetze)**
Untersuchungs-methode	• (In-)formal • Graphisch-deskriptiv	• Formal • Graphen-theoretisch • Simulativ	• (Semi-)formal • Analytisch • Simulativ
Zustands-übergänge	• Statisch	• Dynamisch	• Dynamisch
Zustände		• Wertdiskret	• Wert-kontinuierlich
Verwendungs-zweck	• Wissens-repräsentation	• Ausführbare Analysemodelle	• Ausführbare Analysemodelle

2.2.2 Werkzeuge

In Tabelle 6 sind ausgewählte Petrinetzwerkzeuge gegenübergestellt, die gemäß den Anforderungen an das Analysekonzept als geeignet angesehen wurden. Alle Werkzeuge ermöglichen die Modellierung und Analyse einfacher Petrinetzmodelle. Zwecks der geforderten Kopplungsmöglichkeiten stehen für die meisten Werkzeuge bereits Übergabestellen zur Verfügung wie beispielsweise bei CPNTools die Access/CPN-Umgebung, bei TimeNet der OpenAdaptor oder bei π-Tool eine Anbindung über CORBA.

Tabelle 6: Geeignete Werkzeuge für die Modellierung und Simulation von Petrinetz-Modellen [Quiroga et al. 2014]

	Werkzeuge			
Merkmale	**GreatSPN**	**CPNTools**	**TimeNet**	**π-Tool**
Zeitbehaftete Netze	+	+	+	+
Simulation	+	+	+	+
Hierarchie	-	+	+	+
Modularität	-	+	-	+
Numerische Analyse	+	-	+	+
Erreichbar-keitsgraph	+	+	-	+
Online-Datenüber-gabe	+/-	+	+	+

\- (wird nicht unterstützt bzw. ist ungeeignet), + (wird unterstützt bzw. ist geeignet)

Für die Analysen mittels Petrinetze wurde das Werkzeug π-Tool-gewählt. Das Werkzeug ermöglicht u.a. eine automatisierte Modellerstellung auf Basis mathematischer bzw. textueller Beschreibungen [Diekhake et al. 2016], die auf Informationen aus den wissensbeschreibenden Systemmodellen beruhen. Weiterhin ermöglicht das Werkzeug π-Tool die direkte Anbindung an ein Realsystem, wodurch

Online-Analysen bei der Inbetriebnahme bzw. während des Betriebes durchführbar werden [Diekhake/Schnieder 2013].

SPICE basierte Werkzeuge, die für die Modellierung und Simulation kontinuierlicher elektrischer Schaltungen geeignet sind, sind in Tabelle 7 aufgezeigt. Neben den Grundfunktionen zur Lösung der Differentialgleichungen bieten die Werkzeuge zusätzliche Benutzeroberflächen zur Beschreibung von Schaltbildern sowie z. T. einen erweiterten Funktionsumfang für die Analyse. Eine Werkzeugkopplung kann bei HSPICE mittels OCEAN API, bei NI Multisim über das NI LabVIEW Multisim API Toolkit oder bei PSPICE über eine JSON-, COM-, Direct API- oder die MLPS-Übergabestelle realisiert werden.

Tabelle 7: Geeignete Werkzeuge für die Modellierung und Simulation elektrischer Schaltungen

	Werkzeuge			
	HSPICE	**LTspice**	**NI Multisim**	**PSPICE**
Merkmale				
Analyse-methoden	+	+/-	+	+
Bauteil-begrenzung	+	+	+	+
Graphische Modellierung	-	+	+	+
Bauteil-bibliotheken	+	+/-	+/-	+
Daten-übergabe	+	-	+	+

- (wird nicht unterstützt bzw. ist ungeeignet), + (wird unterstützt bzw. ist geeignet)

Für die Analyse kontinuierlicher elektrischer Schaltungen wurde das Werkzeug PSPICE verwendet. Durch die Nutzung hierarchischer Konzepte und der Möglichkeit der Wiederverwendbarkeit von Modellkomponenten eignet sich der ergänzende Schaltplaneditor OrCAD Capture insbesondere für die Abbildung komplexerer Schaltungen [Diekhake 2013].

3 Systemmodelle zur Wissensrepräsentation

In diesem Kapitel werden die Systemmodelle zur Wissensrepräsentation gemäß dem in Abschnitt 2.2 vorgeschlagenen Analysekonzept beschrieben. Ziel ist es, ein Systemmodell zur Verfügung zu stellen, welches von der abstraktesten Sicht auf den allgemeinen Systembegriff bis hin zur gerätetechnischen Sicht auf ein konkretes Bauelement und seine Kennwerte reichen kann. Dadurch können alle relevanten Informationen für jede der mannigfaltigen Problemstellungen aus Abschnitt 1.2 einheitlich zusammengetragen werden. Die in diesem Kapitel vorgestellten Systemmodelle sind zur Veranschaulichung vorzugsweise für ein beispielhaftes Automatisierungssystem konkretisiert worden, prinzipiell aber allgemein anwendbar.

Auf Basis der beschreibenden Systemmodelle werden für den weiteren Analyseverlauf ausführbare Analysemodelle ableitbar, die wiederum der Ermittlung von noch unbekannten Eigenschaften bzw. Eigenschaftswerten dienen. Um ausführbare Analysemodelle aus dem beschreibenden Systemmodell (automatisiert) zu extrahieren ist auf Eindeutigkeit bei der Einordnung des Systemwissens in das Systemmodell zu achten. Die Anforderungen an einen variablen Abstraktionsgrad und an eine korrekte Einordnung von Systemwissen werden durch folgenden Methoden erfüllt:

- Mit Hilfe zielorientierter Konkretisierungen und (De-)Kompositionen vom Systembegriff werden die Systemgrenzen des zu analysierenden (Sub-)Systems eindeutig definiert. Dadurch kann eine Reduktion der Systemkomplexität herbeigeführt und der Schwerpunkt auf das der jeweiligen Problemstellung zuzuordnende (Sub-)System gelegt werden.

- Die Einführung von Hierarchieebenen ermöglicht sowohl abstrakte als auch konkrete Sichtweisen auf das *System*, seine *Subsysteme* und die *Ressourcen*.

- Die Eindeutigkeit wird mit Hilfe einer Begriffsformalisierung geschaffen, indem die verwendeten Begriffe (hier: *System, Subsystem* oder *Ressource)* über eine Attributhierarchie detailliert beschrieben werden [Schnieder/Schnieder 2012].

In den folgenden Abschnitten werden zunächst die Beschreibungen eines vorhandenen Systems aus systemtheoretischer (Abschnitt 3.1) sowie aus gerätetechnischer Sicht (Abschnitt 3.2) behandelt. Ergänzend wird in Abschnitt 3.3 beschrieben, wie die zu erfüllende Eigenschaften, d. h. die Anforderungen an das (Sub-)System in das Systemmodell, eingeordnet werden können. Im abschließenden Abschnitt 3.4 werden zusammenfassend ganzheitliche Systemmodelle in Form von Beispielen vorgestellt.

3.1 Systemtheoretische Beschreibung

Die theoretische Systemanalyse [Ropohl 2012] ist ein Ansatz zur abstrakten Beschreibung eines Systems. Sie dient dazu, auf der Grundlage systemtheoretischer Erkenntnisse ausreichende Informationen zu gewinnen, um für eine Problemstellung Lösungen zu finden oder Anforderungen an eine Lösung zu formulieren. Typischerweise wird bei der Systemanalyse nach der Top-Down-Methode vorgegangen, indem von einer zuvor festgelegten abstrakten Sicht auf den allgemeinen Systembegriff in der weiteren Bearbeitung eine sukzessive Verfeinerung vorgenommen wird. In diesem Abschnitt wird der allgemeine Systembegriff definiert und exemplarische Konkretisierungen dieses Systembegriffs bis zum im Abschnitt 1.1.1 behandelten Anwendungsbeispiel vorgestellt.

3.1.1 Zentrale Systemeigenschaften

Der weitfassende und schwer definierbare Systembegriff ist nach [Schnieder 1993], [Schnieder/Schnieder 2010] durch folgende Axiome gekennzeichnet:

Strukturprinzip

> Das System besteht aus einer Menge von Elementen, die zueinander in Wechselwirkungen stehen. Das System wird durch eine Systemgrenze abgegrenzt.

Dekompositionsprinzip

> Das System besteht aus einer Menge von Teilen, die wiederum in Unterteile zerlegt werden können. Dadurch ist eine Bildung von Untersystemen und Übersystemen möglich.

Kausalprinzip

> Das Systeme weist eine Logik von Abläufen auf. Damit können eintretende Zustände nur von ihren vorangegangenden Zuständen abhängig sein.

Temporalprinzip

> Durch Zustandsänderungen unterliegt das System zeitlichen Veränderungen.

Für ein *System* lassen sich aus den Systemaxiomen zunächst die wesentlichen Eigenschaften *Struktur*, *Zustand*, *Funktion* und *Verhalten* [Schnieder/Schnieder 2010] ableiten (Abbildung 3.1). Eine nachfolgende Konkretisierung des allgemeinen Systembegriffs führt zu dem eigentlich zu analysierenden System (Abschnitt 3.1.2).

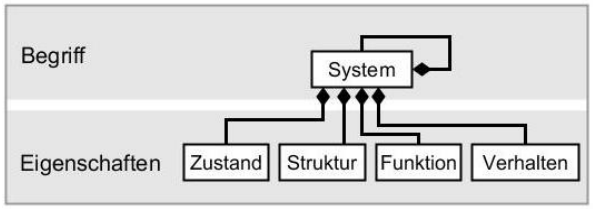


Abbildung 3.1: Allgemeiner Systembegriff und seine Eigenschaften [Schnieder 2010]

3.1.2 Begriffsformalisierung und Konkretisierung des Systembegriffs

Unter Anwendung der Attributhierarchie aus [Schnieder/Schnieder 2012] kann ein Systembegriff durch die Angabe von *Eigenschaften*, *Merkmalen*, *Größen* sowie *Werten und Einheiten* formalisiert werden. Exemplarisch dafür sind in Abbildung 3.2 die Eigenschaften *Struktur* bzw. *Transportprozess* der allgemeingültigen Systembegriffe *System* bzw. *technisches System* eingeführt. Demnach wird die *Struktur* des *Systems* u. a. durch die *strukturelle Komplexität*, d. h. durch die Art und *Anzahl der Systemelemente* sowie die Art und *Anzahl der Kopplungen* zwischen den Systemelementen beschreibbar. Der Eigenschaft des *Transportes* von Informationen, Stoffen oder Energie innerhalb des *technischen Systems* kann die jeweilige Kenngröße des *Durchsatzes* zugewiesen werden.

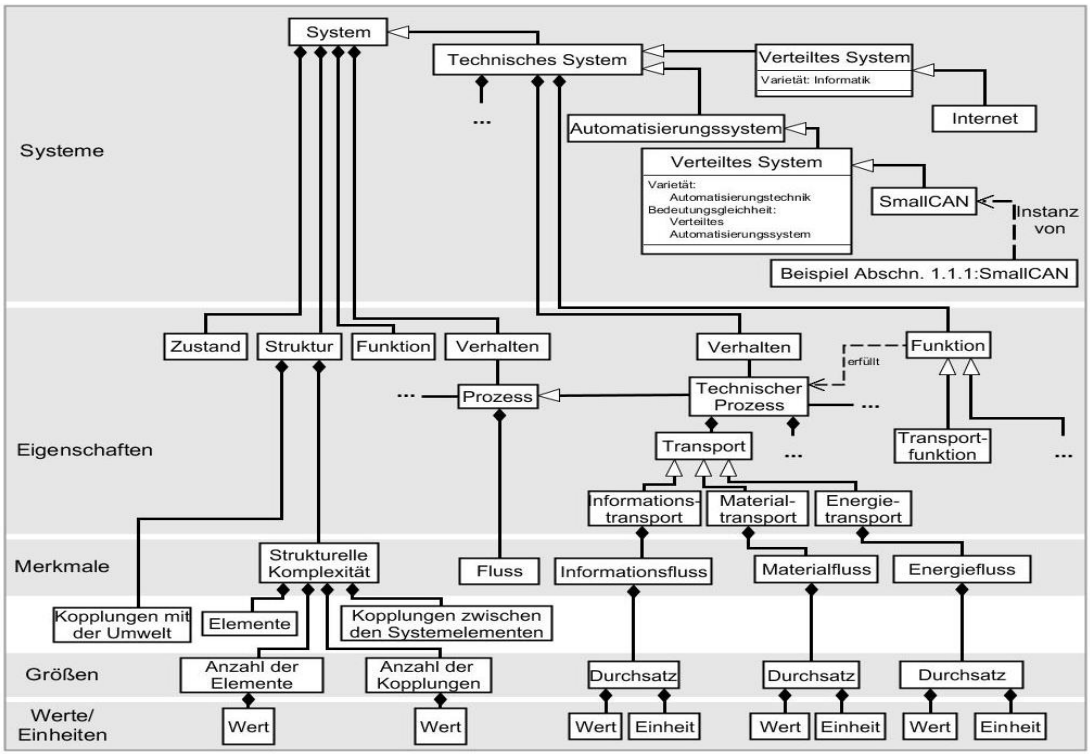


Abbildung 3.2: Konkretisierung vom Systembegriff und exemplarische hierarchische Beschreibungsstruktur am Beispiel der Struktur eines technischen Systems und des Transportprozesses

Ergänzend dazu stellt die Definition von Systemgrenzen eine weitere Aufgabe der Systemanalyse dar. Durch eine Konkretisierung vom allgemeinen Systembegriff kann die Systemgrenze entsprechend der zu betrachtenden Problemstellung festgelegt werden. In Abbildung 3.2 sind auf der Systemebene exemplarische Konkretisierungen des allgemeinen Systembegriffs bis hin zur Darstellung des spezifischen Automatisierungssystems SmallCAN [Schrom 2007] und seiner konkreten Anwendung zur Gebäudeautomatisierung (Abschnitt 1.1.1) aufgezeigt.

Ein *technisches System*, als Konkretisierung des allgemeinen Systembegriffs ist nach [Schneider 1997] ein System, das ausschließlich aus technischen Bestandteilen zusammengesetzt ist. Somit stellt es eine Beschreibung eines auf technischer Weise von Menschen geschaffenen oder gedachten Produktes dar, während der allgemeine Begriff des Systems nicht zwangsweise ein Produkt repräsentieren muss. Bezugnehmend auf die vererbten zentralen Eigenschaften eines Systems weist ein technisches System als systemtheoretisches Modell eines Produktes eindeutige Systemgrenzen auf. Der *Zustand* des Systems kann zu jedem Zeitpunkt angegeben werden und zu einem gewissen Grad sogar vorhersagbar sein. Die *Funktion* ist mit der Erfüllung des technischen Prozesses, der sich nach der Definition aus der [DIN IEC 60050-351] aus Vorgängen der Verarbeitung, dem Transport und der Speicherung von Stoffen, Energien und Informationen in einer technischen Anlage zusammensetzt, vorgegeben. In [Schnieder 1993: 18] wird der allgemeine Begriff des Prozesses als Teilmenge der axiomatischen Eigenschaften eines Systems definiert, in Abbildung 3.2 angedeutet durch die offene Relationskante der Klasse des Prozesses. Das V*erhalten* eines technischen Systems ist mathematisch beschreibbar und hängt unmittelbar mit den Material- Energie- und Informationsflüssen (z. B. Materialdurchsatz $\lambda \left[\frac{\text{Stück}}{\text{Tag}}\right]$, elektrische Leistung P [W] oder Datenrate b $\left[\frac{\text{bit}}{\text{s}}\right]$) zusammen. Durch die Steuerbarkeit und Regelbarkeit ist ein technisches System im Allgemeinen stabilisierbar. Die *Struktur* eines technischen Systems wird mit Angaben über die strukturelle Komplexität beschreibbar. Die strukturelle Komplexität setzt sich aus den Systemelementen und deren Beziehungen zueinander, d. h. der Kopplung der Systemelemente, zusammen.

Ein *Prozessautomatisierungssystem* besteht nach [Lauber/Göhner 1999] aus dem nicht-technischen Untersystem *Mensch* und den technischen Untersystemen *Automatisierungsobjekt* und *Automatisierungssystem*. Bedingt durch die Zielvorgaben übernimmt der Mensch die Leitung und Bedienung des technischen Prozesses. Das Automatisierungsobjekt ist dabei das technische System, welches automatisiert werden soll. Es wird aus ressourcentechnischer Sicht auch als technische Anlage bezeichnet, auf der der technische Prozess abläuft [Polke 1994: 26]. Als Automatisierungssystem werden diejenigen Einrichtungen bezeichnet, die zur Automatisisierung des

Automatisierungsobjektes beitragen. Es beinhaltet die Rechner- und Kommunikationssysteme sowie die Schnittstellen zum technischen Prozess (Sensoren und Aktoren). Im Vergleich zum technischen System sind im Automatisierungssystem lediglich die Umwandlung, der Transport und die Speicherung von Informationen und ggf. der Transport von Energie, sofern selbstversorgende Systeme betrachtet werden, vorherrschend. Daher stellt die Assoziation zwischen dem Automatisierungssystem und dem technischen System einen prototypischen Vererbungstyp dar (vgl. Abbildung 3.2), d. h. das vererbte Wissen kann optional oder zumindest durch gegenteilige Informationen überschreibbar sein [Helbig 2008: 47] und ist damit nicht zwingend vollumfassend zu übernehmen.

Aus den Definitionen nach [Coulouris et al. 2002] und [Tanenbaum/van Steen 2008] lässt sich ableiten, dass *verteilte Systeme* als eine Zusammenfassung räumlich verteilter Komponenten aufgefasst werden können, die als ein zusammenhängendes System erscheinen. Der Begriff des verteilten Systems ist, wie der Systembegriff selbst, sehr abstrakt und findet eher in der Begriffswelt (Varietät) der Informatik Anwendung. [Tari/Bukhres 2001] definierten verteilte Systeme als Sammlung autonomer Computer, die über ein Netzwerk verbunden sind. Das Netzwerk kann im Kontext der Computervernetzung mit der LAN-Technologie (Ethernet) gleichgesetzt werden. Unter Berücksichtigung spezifischer Randbedingungen, beispielsweise zeitlicher Restriktionen und Zuverlässigkeitsanforderungen auf der Feldebene ist eine Unterscheidung zwischen den ethernetbasierten *verteilten (Computer-)Systemen* und den *verteilten Automatisierungssystemen* zu treffen. Weiter unterscheiden sich beide Begriffe in den zu ergänzenden Funktionen zur Zustandserfassung und -beeinflussung eines technischen Prozesses als eine wesentliche Aufgabe eines Automatisierungssystems. Es bleibt zu erwähnen, dass ein verteiltes Automatisierungssystem auch auf Ethernet basieren (z. B. EtherCAT) bzw. als abgeschlossenes Modul über einen Zugang in ein ethernetbasiertes verteiltes (Computer-)System eingebettet werden kann, beispielweise empfiehlt das BSI die Integration eines HAN in die Umgebung eines *Smart Grids* [BSI TR-03109-1:2013] [Eichelberg 2010].

Ein spezifisches verteiltes Automatisierungssystem ist *SmallCAN*. Eine konkrete anwendungsbezogene Implementierung bzw. Instanz von SmallCAN stellt das *Anwendungsbeispiel aus Abschnitt 1.1.1* dar.

3.1.3 (De-)Komposition von Subsystemen

Nach der Festlegung der Systemgrenzen durch eine konkretisierende Sicht auf den Systembegriffs kann das zu analysierende (Sub-)System durch (De-)Kompositionsprozesse variabel verfeinert bzw. wieder vergröbert abgebildet werden. Eine Dekomposition kann dabei z. B. gemäß der Partitionierungsprinzipien nach [Schnieder 2010] erfolgen. Bezogen auf eine *räumliche Partitionierung* wird das Automatisierungssystem z. B. in 2 bis *n* räumlich verteilte Busteilnehmer aufgeteilt. Entsprechend vorherrschender Prozesse kann das System ebenso nach dem Kriterium der *funktionalen Partitionierung* gegliedert werden. Demnach werden die Aufgaben der Zustandsbeeinflussung, der Zustandserfassung und der Verarbeitung dem Anwendungssystem zugeordnet und das Kommunikationssystem gemäß dem ISO-OSI-Referenzmodell [Zimmermann 1980] in Funktionsschichten der Kommunikationsaufgabe aufgeteilt. Die physikalische Kopplung der Kommunikationssysteme untereinander erfolgt über das Übertragungskanalsystem, welches nach dem ISO-OSI-Referenzmodell nicht Teil des Kommunikationssystems ist. In Anlehnung an eine *technologische Partitionierung* erfolgt eine Dekomposition durch eine Unterscheidung zwischen Hardware- und Softwarerealisierungen. Gemäß der drei vorgestellten Partitionierungsprinzipien wird in Abbildung 3.3 die Aufteilung eines verteilten Automatisierungssystems in seine Subsysteme aufgezeigt.

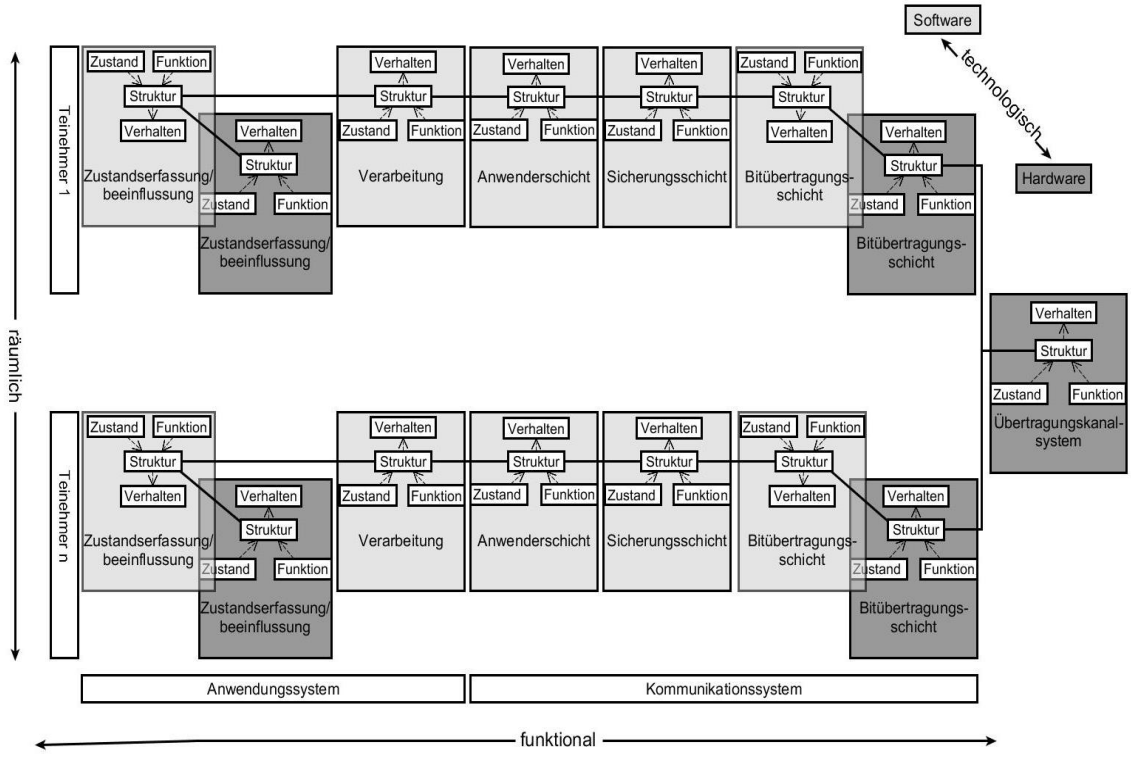


Abbildung 3.3: Darstellung der dekomponierten Subsysteme eines verteilten Automatisierungssystems

Anwendungssystem

Die in Abbildung 3.3 skizzierten einzelnen Subsysteme lassen sich durch geeignete Kompositionen wieder zu neuen übergeordneten (Sub-)Systemen zusammenfassen und anschließend mittels der Attributhierarchie wieder detailliert beschreiben. Exemplarisch ist eine solche Komposition mit nachgeführter Begriffsattribuierung in Form eines Klassendiagrammes in Abbildung 3.4 dargestellt.

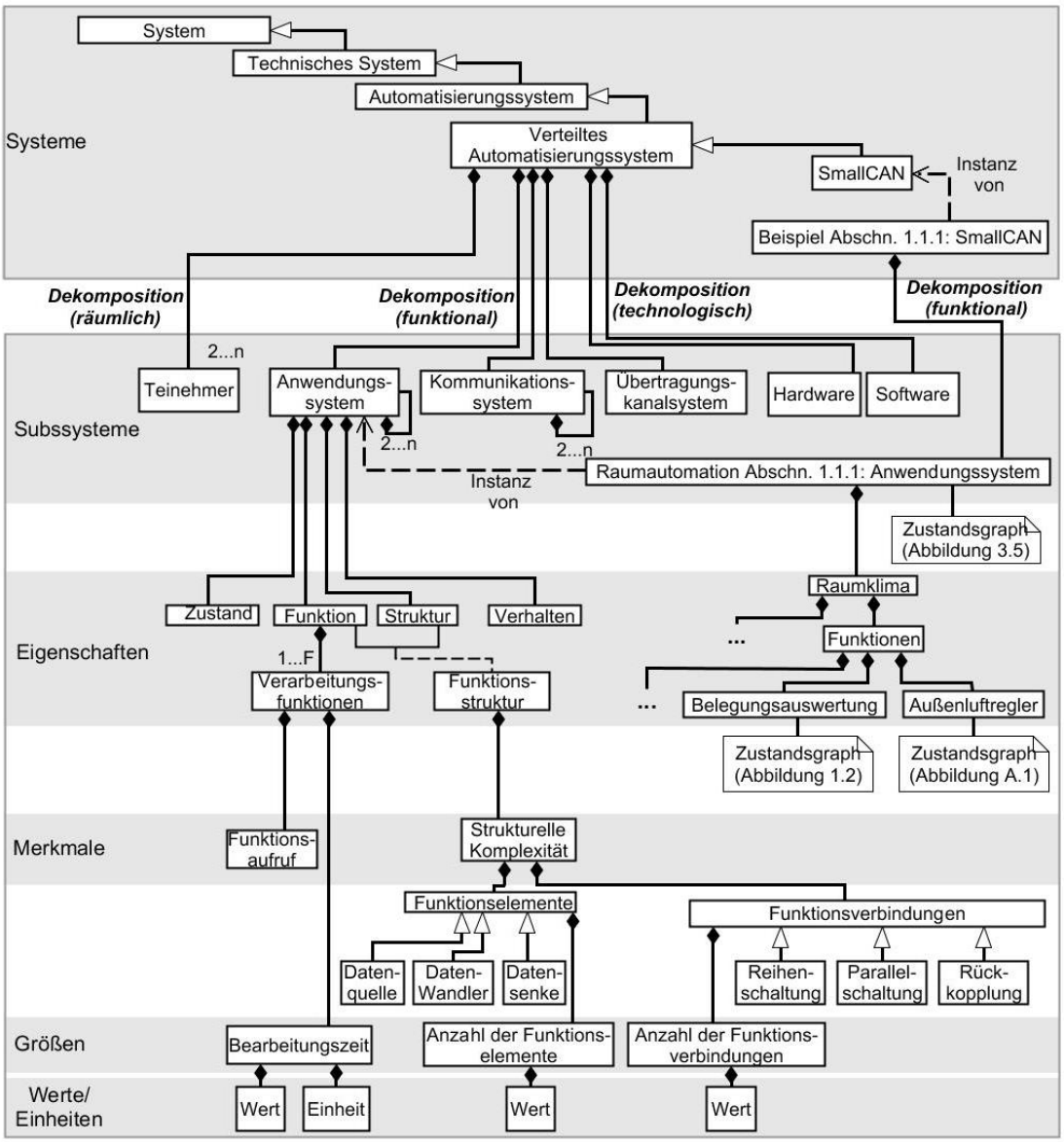


Abbildung 3.4: Ausschnitt aus dem Hierarchieschema zur Beschreibung des funktionalen Anwendungssystems eines verteilten Automatisierungssystems und der Raumautomation als dessen Konkretisierung

In der Abbildung werden die Eigenschaften des gesamten *Anwendungssystems*, bestehend aus sämtlichen (2 bis *n*) einzelnen *Anwendungssystemen*, aus einer rein funktionalen Sichtweise heraus beschrieben. In dieser funktionsorientierten Darstellung

bleiben die räumlichen sowie die technologischen Partitionierungsaspekte zunächst unberücksichtigt.

Als Komposition aller 2 bis *n* Anwendungssysteme besteht das gesamte Anwendersystem aus 1 bis *F Verarbeitungsfunktionen*. Diese können z. B. einfache Grundfunktionen, Ansteuerungsfunktionen für Sensorik und Aktorik, Zeitfunktionen, Skalierungsfunktionen, Vergleichsfunktionen, Logikverknüpfungen, Ablaufsteuerungen oder Anzeigefunktionen sein, die u. a. durch ihre *Bearbeitungszeiten* und ihr *Aufrufverhalten* charakterisiert sind. Nach UML/SysML-Notation stellt die *Funktionsstruktur* eine Assoziationsklasse dar, die die Assoziationsattribute der Klassen *Funktion* und *Struktur* beinhaltet. Die Verarbeitungsfunktionen können innerhalb der Funktionsstruktur des Anwendungssystems demnach als *Funktionselemente* aufgefasst werden. In Anlehnung an die Strukturbeschreibung eines allgemeinen Systems (Abbildung 3.2) werden die Kopplungen zwischen den Systemelementen als *Funktionsverbindungen* bezeichnet. Als Typen der Funktionselemente können *Datenquellen*, *-senken* oder *-wandler* definiert werden, die über die Funktionsverbindungen in Form von *Reihen,- Parallelschaltungen* und *Rückkopplungen* miteinander verbunden sind.

Bezugnehmend auf das Anwendungsbeispiel aus Abschnitt 1.1.1 kann die realisierte *Raumautomation*, die schematisch und stark vereinfacht durch den Zustandsgraphen aus Abbildung 3.5 dargestellt ist, als eine Instanz des funktionalen Anwendungssystems aufgefasst werden. Durch die im Systemmodell aufgeführten Verweise kann ein Bezug sowohl zu den Zustandsgraphen der Raumautomation (Abbildung 3.5) als auch zu einzelnen verarbeitenden Funktionen wie beispielsweise der *Belegungsauswertung* (Abbildung 1.2) oder der *Außenluftregelung* (Abbildung A.1) hergestellt werden.

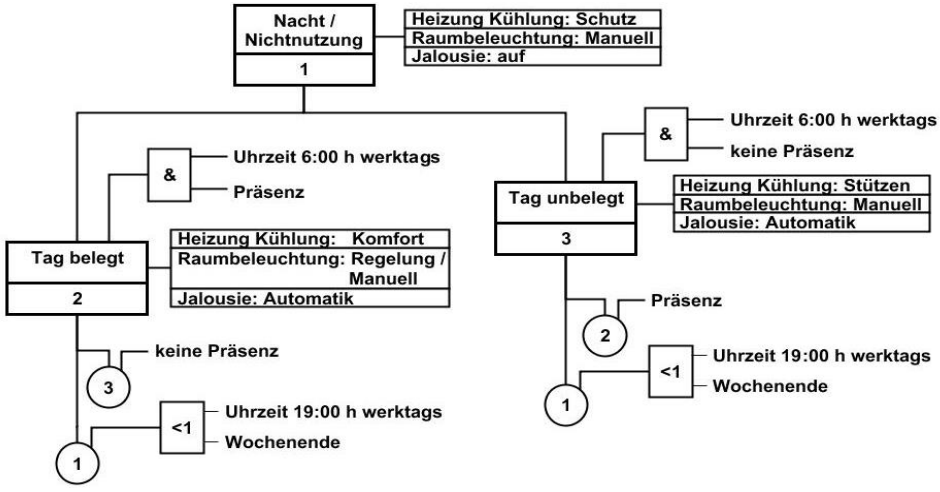


Abbildung 3.5: Abstrahierter Zustandsgraph der implementierten Raumautomation (in Anlehnung an die Richtlinie [VDI 3814:2009])

SmallCAN-Bitübertragungssystem

Durch eine kombinierte Anwendung der Partitionierungsprinzipien lassen sich Subsysteme anderweitig (de-)komponieren. Abbildung 3.6 zeigt das Hierarchieschema für das *Bitübertragungssystem* des verteilten Automatisierungssystems SmallCAN als Komposition sämtlicher *Bitübertragungsschichten (HW)* des Kommunikationssystems, die über das *Übertragungskanalsystem* physikalisch miteinander gekoppelt sind. Die Darstellung ist sowohl von der technologischen Sicht auf die Hardware als auch von räumlichen Aspekten geprägt. Eine zutreffende Beschreibung des *Zustandes* des Bitübertragungssystems kann über die Angabe des dominanten bzw. rezessiven *Signalpegels* auf dem Übertragungskanal erfolgen. Die vorherrschende *Funktion* des Bitübertragungssystems ist die *Signalübertragung*, die den Informationstransport ausführt. Der *Bitstrom* wiederum ist ein Merkmal der Signalübertragung und kann über die Größe des *Durchsatzes*, d. h. in diesem Fall über die *Bitrate* angegeben werden. Das *Verhalten* des Bitübertragungssystems ist im Wesentlichen durch den physikalischen Signalverlauf gekennzeichnet, für den exemplarisch die Größen des *Spannungsabfalls* und der *Einschwingzeit* angegeben sind. Die *Struktur* des Bitübertragungssystems lässt sich in Form einer physikalischen Mischtopologie aus *Sternen*, *Bussen* und *Bäumen* abbilden, wobei eine *Topologie* im Wesentlichen durch die folgenden systemtheoretischen Größen gekennzeichnet ist:

Anzahl Knoten

Analog den Elementen eines Systems (Abbildung 3.2) besteht eine Systemtopologie aus N Knoten. Bei der Betrachtung der physikalischen Topologie von SmallCAN entspricht jeder Knoten einem Kommunikations-Baustein, einer datenstromeinspeisenden Zentraleinheit oder einem Anschluss- bzw. Abzweigpunkt für Datenleitungen.

Anzahl Kanten

Analog den Kopplungen zwischen den Elementen eines Systems (Abbildung 3.2) besteht eine Systemtopologie aus K Kanten. Für die physikalische Topologie entspricht jede Kante einer direkten Verdrahtungsleitung oder einer Busleitung.

Durchmesser

Der Durchmesser einer Topologie beschreibt die maximale direkte Entfernung zwischen zwei Knoten im System.

Grad

Der Grad einer Topologie gibt die maximale Anzahl der Verbindungen pro Knoten an.

Bisektionsweite

Die Bisektionsweite gibt die minimale Anzahl von Kanten an, die durchschnitten werden müssen, um ein System mit *N* Knoten in zwei Systeme mit jeweils *N*/2 Knoten zu teilen.

Kantenkonnektivität

Die Kantenkonnektivitat gibt die minimale Anzahl von Verbindungen an, die durchtrennt werden muss, damit das System als solches nicht mehr funktionstüchtig ist.

In Tabelle 8 werden die Kennwerte verschiedener Grund-Topologien gegenübergestellt. Da das Bitübertragungssystem gemäß Abbildung 3.6 auch aus technologischer Sicht skizziert ist, lässt sich ein Bezug zwischen den theoretischen Kenngrößen und den physikalischen Qualitätsmerkmalen berücksichtigen. Beispielsweise stellt der Grad der Netztopologie ein Maß für die Geräte- und Verlegekosten dar. Der Durchmesser, die Kantenkonnektivität sowie die Bisektionsweite hängen mit der Zuverlässigkeit und Leistungsfähigkeit des Systems zusammen, wobei die Bisektionsweite und die Kantenkonnektivität möglichst groß und der Durchmesser möglichst klein sein sollten. Neben den topologischen Kenngrößen sind für die Beeinflussung der Qualitätsmerkmale auch praktische Kriterien von Relevanz. Beispielsweise kann es bei einer Sternverkabelung zum Übersprechen zwischen individuell angesteuerten Datenleitungen und damit zu einer Beeinträchtigung der Signalqualität kommen. Aufgrund der notwendigen Verarbeitung der Datensignale in jedem Transceiverbaustein steigt innerhalb einer Ringstruktur die Reaktionsverzögerung proportional zur Anzahl der Teilnehmer an, was zu signifikanten Leistungseinbußen führen kann. Auch der Wartungsaufwand ist je nach eingesetzter Topologie unterschiedlich hoch. Insbesondere für die Bustopologie erweist sich die Fehlersuche u. a. aufgrund ergänzend einzusetzender Abzweig- bzw. Verbindungsstücke als schwierig.

Tabelle 8: Kenngrößen von physikalischen Netztopologien und deren Beeinflussung auf ausgewählte Qualitätsmerkmale

	Topologie			
	Stern	**Ring**	**Linie/Bus**	**Bus**
Theoretische Kenngrößen				
Anzahl Knoten	*N*	*N*	*N*	*N*+ *N*/2
Anzahl Kanten	*N*-1	*N*	*N*-1	*N*-1+ *N*/2
Durchmesser	2	*N*/2	1	1
Grad	*N*-1	2	2	3
Bisektionsweite	*N*/2	2	1	1
Kantenkonnektivität	1	2	1	1
Praktische Kriterien (Beispiele)	Übersprechen auf Leitungen	Skalierbar-keit	Fehlersuche	
Qualitätsmerkmale	**Beeinflussung durch theoretische Kenngrößen/ Praktische Kriterien**			
Verlässlichkeit	-	+ / -	+ / -	+ / -
Leistung	+	-	+	+
Wirtschaftliche Aspekte	-	-	+	-

- (negativ beeinflusst), + (positiv beeinflusst)

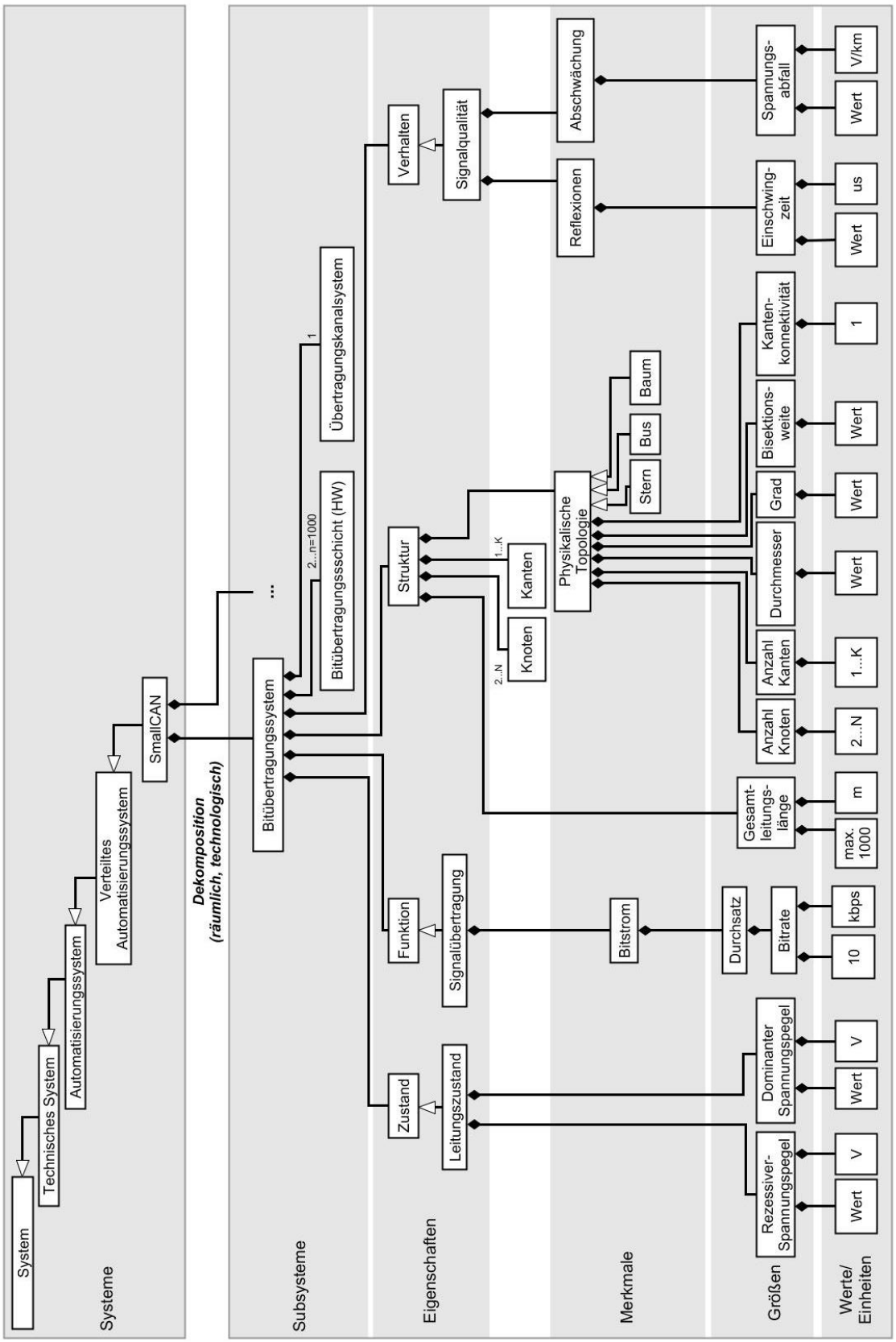


Abbildung 3.6: Ausschnitt aus dem Hierarchieschema zur Beschreibung des Bitübertragungssystems am Beispiel von SmallCAN

SmallCAN-Datenübertragungssystem

Im Folgenden wird eine Komposition des behandelten *Bitübertragungssystems* mit den verbleibenden ISO-OSI Schichten des *Kommunikationssystems* vorgestellt. Diese Komposition kann als *Datenübertragungssystem* aufgefasst werden, dessen hierarchische Beschreibung ausschnittsweise in Abbildung 3.7 dargestellt ist. Das Teilsystem ist sowohl von funktionalen als auch von technologischen und räumlichen Aspekten geprägt.

Bei dem Datenübertragungssystem richtet sich das Hauptaugenmerk auf eine fehlerfreie Kommunikation, so dass dessen *Zustand*s- oder Verhaltensbeschreibung sinnvollerweise über die Angabe von *Fehlerraten* zu beschreiben ist. Hinsichtlich der *Funktion* des Datenübertragungssystems repräsentiert die *Datenübertragung* in Analogie zur Signalübertragung im Bitübertragungssystem den Informationstransport, der über den *Telegrammfluss* beschrieben bzw. über den *Durchsatz* in Form einer *Telegrammrate* angegeben werden kann. Zur Beschreibung der *Struktur* des Datenübertragungssystems ist neben der *physikalischen* auch die *logische Topologie* zu berücksichtigen, für die ebenfalls die in Tabelle 8 eingeführten theoretischen Kenngrößen herangezogen werden können. Im Falle der logischen Topologie eines Busses weist der *Durchmesser* einen Wert von 1 auf, d. h. es ist ein direkter Nachrichtenaustausch zwischen den einzelnen Kommunikationssystemen ohne Routing möglich. Die *Bisektionsweite* als ein Maß für die Mehrfachbelegung einer Leitung beträgt für den Bus ebenfalls 1, d. h. es steht nur ein Übertragungskanal für den Datentransport zur Verfügung. Die *Anzahl der Knoten* einer logischen Topologie entspricht der Anzahl der an den Kommunikationsprozessen beteiligten Geräte, d. h. sämtlichen Busteilnehmern des verteilten Automatisierungssystems. Für das System SmallCAN kann demnach die logische Topologie aus maximal N=1000 Knoten bestehen. Jeder *Knoten* bzw. Busteilnehmer kann bis zu 16 Telegramme empfangen und bis zu 16 Telegramme senden. Entsprechend können in einem maximal ausgeprägten SmallCAN System bis zu K=32000 logische Verknüpfungen hergestellt werden, die in dem Systemmodell aus Abbildung 3.7 über die *Anzahl der Kanten* beschrieben sind. Das *Verhalten* des Datenübertragungssystems wird vorzugsweise über die *Busauslastung* oder die *Hochlaufzeit* nach einem Neustart charakterisiert.

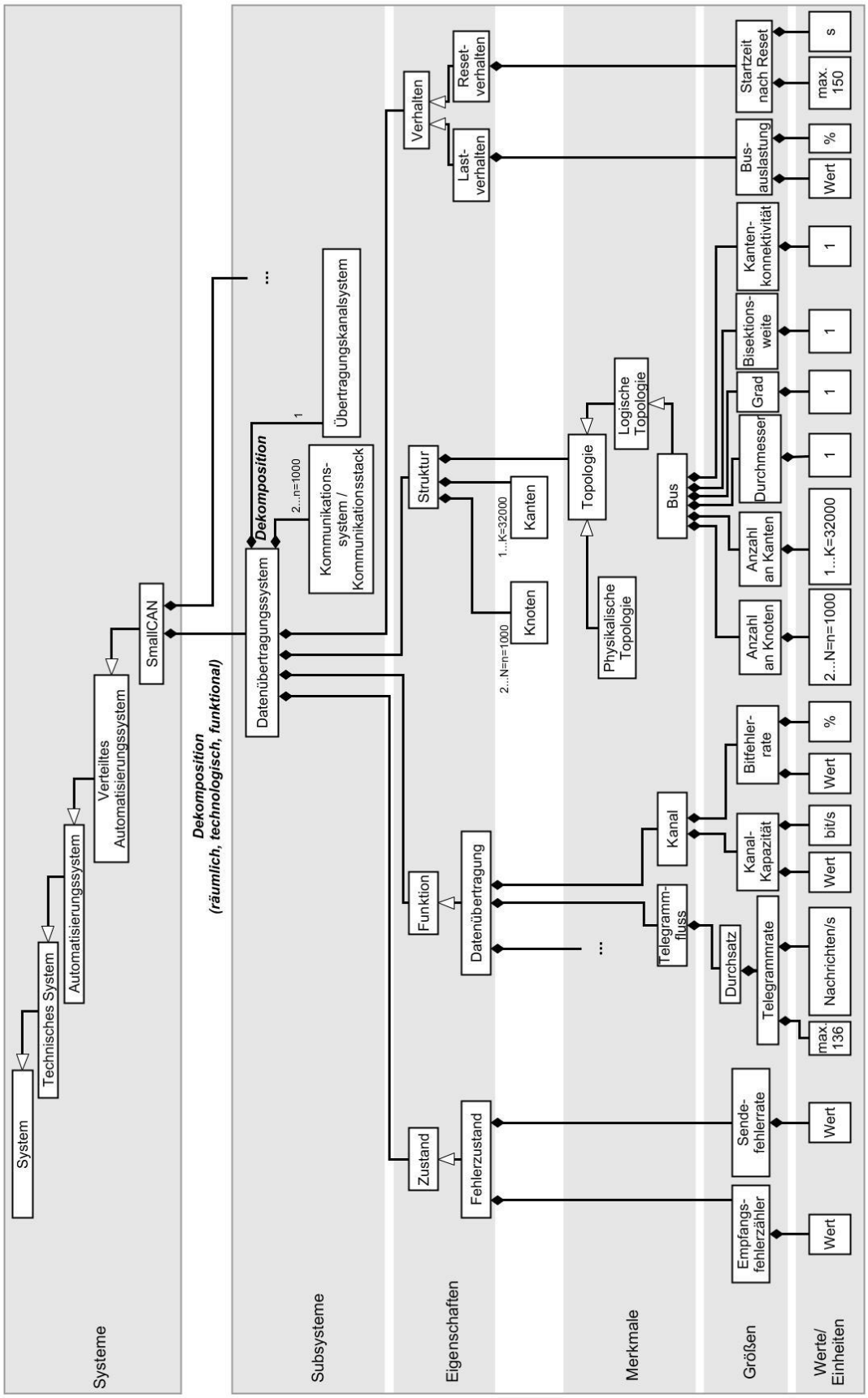


Abbildung 3.7: Ausschnitt aus dem Hierarchieschema zur Beschreibung des Datenübertragungssystems am Beispiel von SmallCAN

3.2 Gerätetechnische Beschreibung

Durch die Dekomposition basierend auf technologischen Aspekten, durch die Konkretisierung in Richtung umsetzbarer Systeme (Spezialisierung, Instanziierung) und bzw. oder durch die Detailierung bis hin zu einzelnen Systemelementen bekommen die zu beschreibenden (Teil-)Systeme einen zunehmend realen Charakter. Dadurch lassen sie sich nur noch bedingt aus einer abstrakten systemtheoretischen Sichtweise mittels zentraler Systemeigenschaften heraus beschreiben. Eine zweckmäßigere Darstellung von Gerätekomponenten, deren Beschreibung ebenfalls Teil der klassischen Systemanalyse ist, kann durch (ergänzende) Angabe von Qualitätseigenschaften erzielt werden.
Entsprechend ist das bisher systemtheoretisch entwickelte Modell um eine Betrachtungsebene zu erweitern, die Ressourcen (ausschließlich) über ihre physikalischen Eigenschaften beschreibt. Eine Ressource kann als Träger einer Systemfunktion aus einer abstrakteren Systemebene verstanden werden. Der Übergang zwischen der Systemfunktion aus der systemtheoretischen Sicht und dem Ressourcenbegriff aus der gegenstandorientierten Sicht wird in [Schnieder 2010] als *Ressourcenallokation* bezeichnet.

In Anlehnung an das Anwendungsbeispiel zur Raumklimatisierung aus Abschnitt 1.1.1 zeigt Abbildung 3.8 ausschnittsweise das Hierarchieschema zur formalisierten Beschreibung des Anwenderprogrammes *AUSLG* (Softwarekomponente bzw. *Programm*) sowie des Anwendungsmoduls *ANMUL* (Hardwarekomponente bzw. *Anwendungsmodul*) als Ressourcen zur Erfüllung der Systemfunktion einer Außenluftregelung (Assoziation zwischen Ressource und Funktion gemäß der Allokationsnotation aus SysML [Alt 2012: 60]). Die Systemfunktion der *Außenluftregelung* ist Teil der *Raumklimatisierung*, die wiederum neben den Funktionen zur *Beschattung* und *Beleuchtung* als eine Funktion zur *Raumautomation* aufgefasst werden kann (vgl. Abbildung 3.5). Für das Anwenderprogramm als softwaretechnische Ressource lassen sich u. a. Angaben über den Quellcode (z. B. *Anzahl von Codezeilen*) oder den notwendigen Speicherbedarf (z. B. *Speichergröße*) machen. Für die Beschreibung des Anwendungsmoduls als hardwaretechnische Ressource sind in Abbildung 3.8 exemplarisch der *Energiebedarf*, die *Kosten* und die *geometrischen Eigenschaften* skizziert. Der Systemfunktion zugehörige Zustandsgraph, der Quellcode des Anwenderprogramms und das Platinenlayout des erstellten Anwendungsmoduls sind im Systemmodell referenziert und können dem Anhang A entnommen werden.

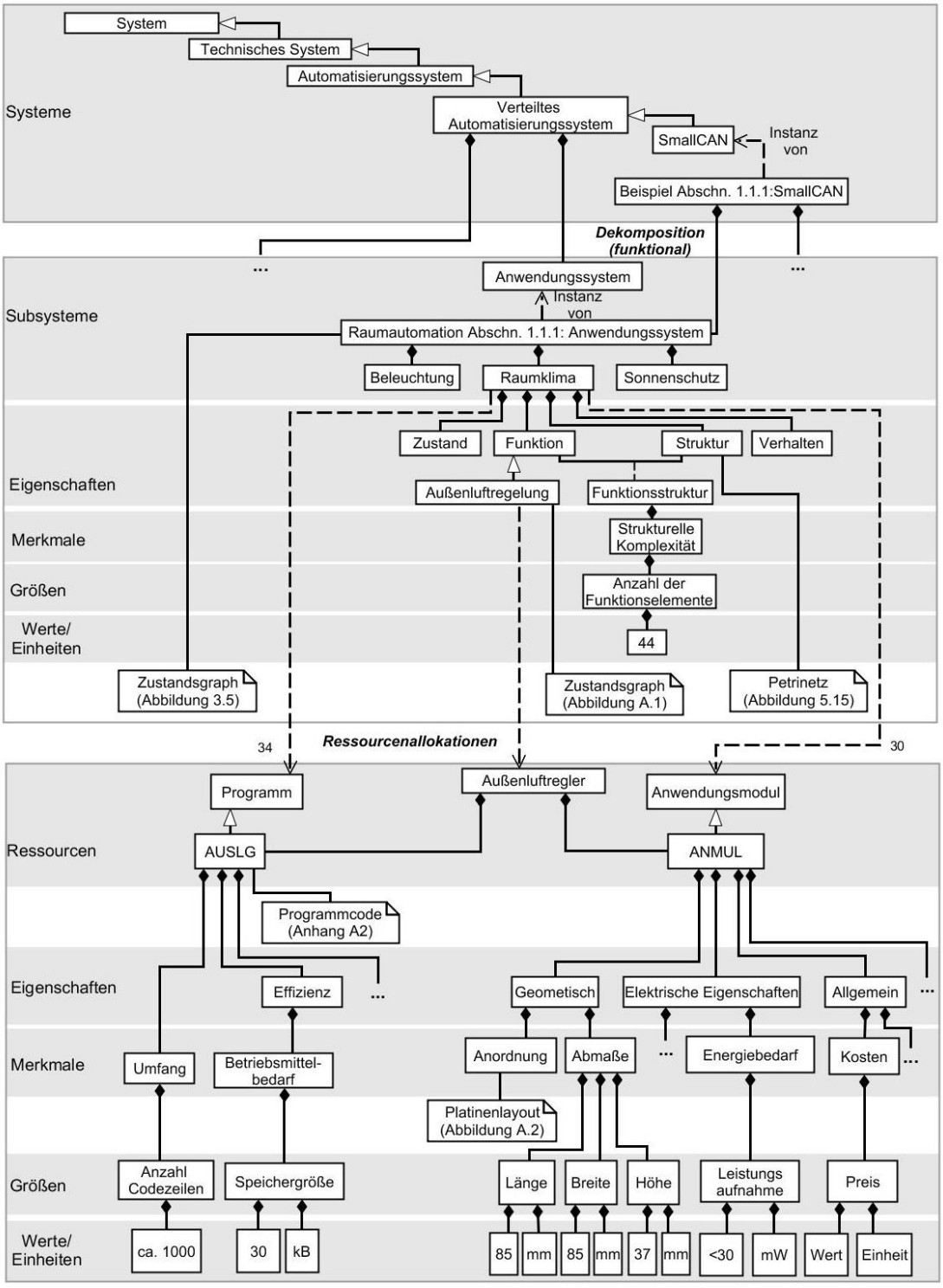


Abbildung 3.8: Ausschnitt aus dem Hierarchieschema zur Beschreibung von SmallCAN-Ressourcen zur Realisierung einer Außenluftregelung

Bezugnehmend auf das Datenübertragungssystem zeigt Abbildung 3.9 die formalisierte Beschreibung eines einfachen Telefonkabels (*J-Y(ST)Y 2x2x0.6*), welches für das SmallCAN-System als *Busmedium* zur Anwendung kommt. Im dargestellten

Hierarchieschema sind die wesentlichen Eigenschaften und Kenngrößen dieser Ressource zusammengefasst.

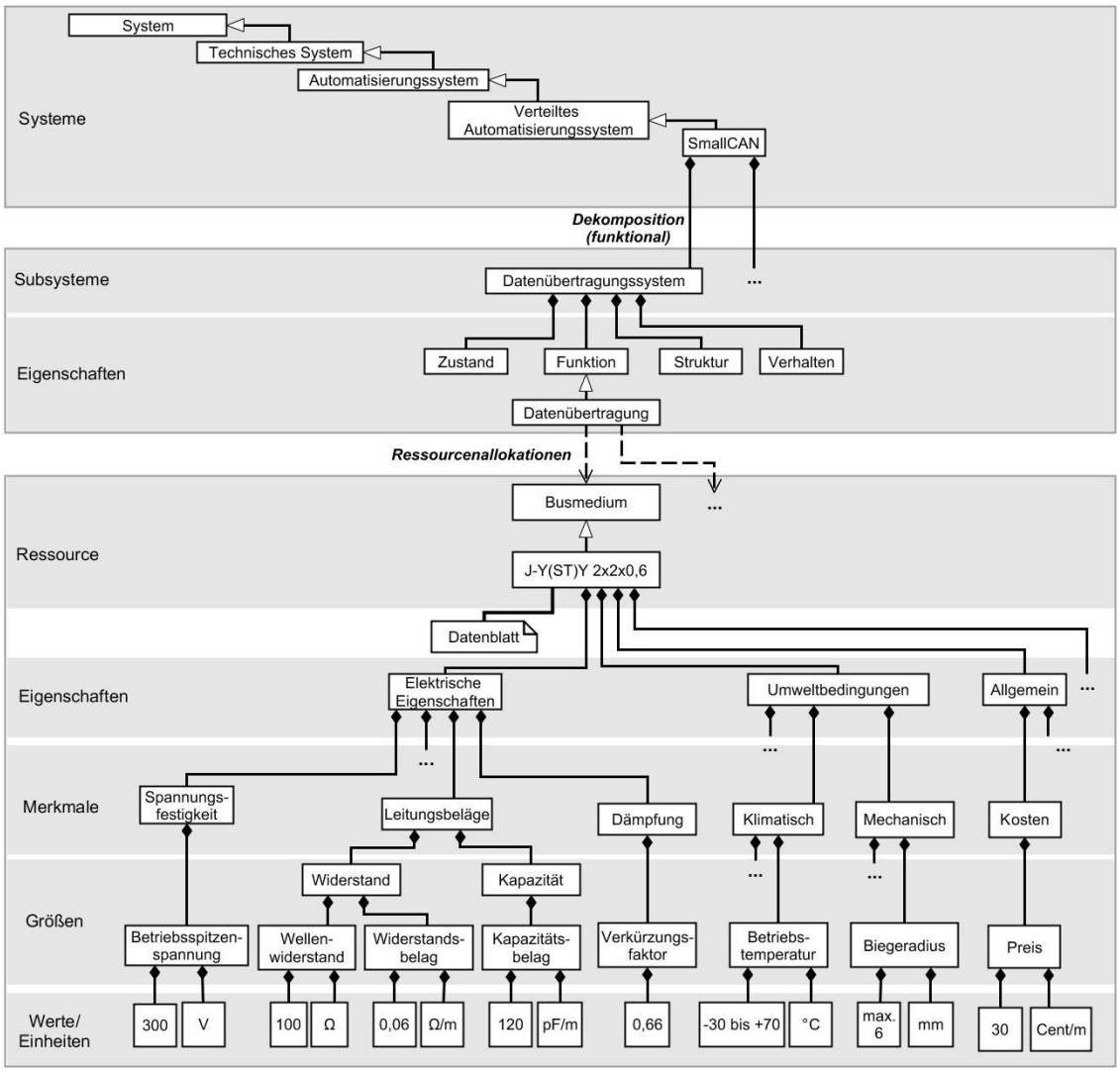


Abbildung 3.9: Ausschnitt aus dem Hierarchieschema zur Beschreibung einer Ressource am Beispiel eines einfachen Telefonkabels als Busmedium

3.3 Anforderungsdefinition

In den vorherigen Abschnitten sind die bisher bekannten Systemeigenschaften aus systemtheoretischer sowie aus gegenstandsorientierter Sicht ermittelt und formalisiert worden. In Bezug auf den weiterführenden Entwicklungsprozess und insbesondere im Kontext der vorliegenden Problemstellungen (Abschnitt 1.2) sind in der Systemanalyse aber auch Anforderungen zu formulieren, d. h. diejenigen Eigenschaften, die das System im Verlauf der Weiterentwicklung und Implementierung noch erfüllen muss. Ein methodisches Vorgehen zur Anforderungsdefinition ist bereits in [Kupke 2007]

vorgestellt und für ein funkbasiertes Kommunikationssystem durchgeführt worden. Das dort vorgeschlagene Vorgehen sieht zunächst eine Untergliederung der Hauptanforderungen vor, so dass sich sowohl allgemeine systemische als auch spezifische Anforderungen an Teilsysteme oder Systemelemente angeben lassen. Weiterhin werden die zu erfüllenden Eigenschaften nach *betrieblichen, technischen* und *funktionalen* Anforderungen klassifiziert und gemäß der Attributhierarchie strukturiert. Als Ergebnis liefert die Methodik eine zusammenfassende Darstellung der Anforderungen an das Gesamtsystem, dessen Teilsysteme und Systemelemente in Form einer Anforderungsmatrix. Das in [Kupke 2007] beschriebene Vorgehen zur Anforderungsdefinition ähnelt der diesem Kapitel zugrundeliegenden Methodik zur Systembeschreibung, so dass eine Integration beider Vorgehen naheliegt. In Abbildung 3.10 ist die Synthese beider Vorgehen dargestellt. Es ist zu berücksichtigen, dass sich insbesondere durch die Ergänzung qualitätsbezogener Systemanforderungen innerhalb einer systemtheoretischen Darstellung diese mit der gerätetechnischen Betrachtungsebenen vermengt, wodurch eine konglomerierte Sichtweise sowohl auf das (Sub-)System als auch auf die Ressource entsteht. Entsprechend ist eine korrekte Kennzeichnung einer solchen vereinigten Ebene zu wählen (*System / Ressource*). Weiterhin ist eine Ebene zur Einordnung von Sollwerten (*Sollwerte / Einheiten*) eingeführt sowie eine Möglichkeit geschaffen worden, um auf die den Anforderungen entsprechende Normlandschaft zu verweisen (*Normen / Richtlinien*).

In Abbildung 3.10 sind diejenigen Anforderungen an das Gesamtsystem und dessen Teilsysteme dargestellt, deren Erfüllung mit Hilfe der im weiteren Verlauf der vorliegenden Arbeit vorgestellten Analysen überprüft wurde. Auf der Eigenschaftsebene (*Anforderungen / Eigenschaften*) sind diese Anforderungen als noch zu erfüllende Eigenschaften dunkelgrau hinterlegt.

Um eine ausreichende Signalqualität auf den Busleitungen des Bitübertragungssystems zu gewährleisten, ist für die derzeitige Implementierung des Systems SmallCAN eine *Einschwingzeit* unterhalb von 11,9 µs gefordert [Schrom 2003: 188]. Innerhalb dieser Zeit ist eine *Schwellenspannung* von 13 V zu erreichen. Weiterhin ist ein tolerierbarer *Spannungsabfall* von maximal 7 V pro Ader nicht zu überschreiten [Schrom 2003: 89]. Auf Basis dieser Anforderungsattribute wird in Abschnitt 4.1 die Signalqualität für verschiedene, teils umfangreich ausgeprägte physikalische Topologien bewertet. In diesem Zusammenhang stellt die *elektromagnetische Verträglichkeit* eine *betriebliche Anforderung* für das Gesamtsystem dar, die mit Einhaltung entsprechender Richtlinien und Normen zu erfüllen ist. Demnach sind für das System SmallCAN die Fachgrundnormen [DIN EN 61000-6-2] für die *Störfestigkeit* im Industriebereich und [DIN EN 61000-6-3] für die *Störaussendung* in Wohn- und Geschäftsbereiche gewählt

worden. Auf die durchgeführten Untersuchungen hinsichtlich der elektromagnetischen Verträglichkeit wird in den Abschnitten 4.2 und 4.3 detailliert eingegangen.

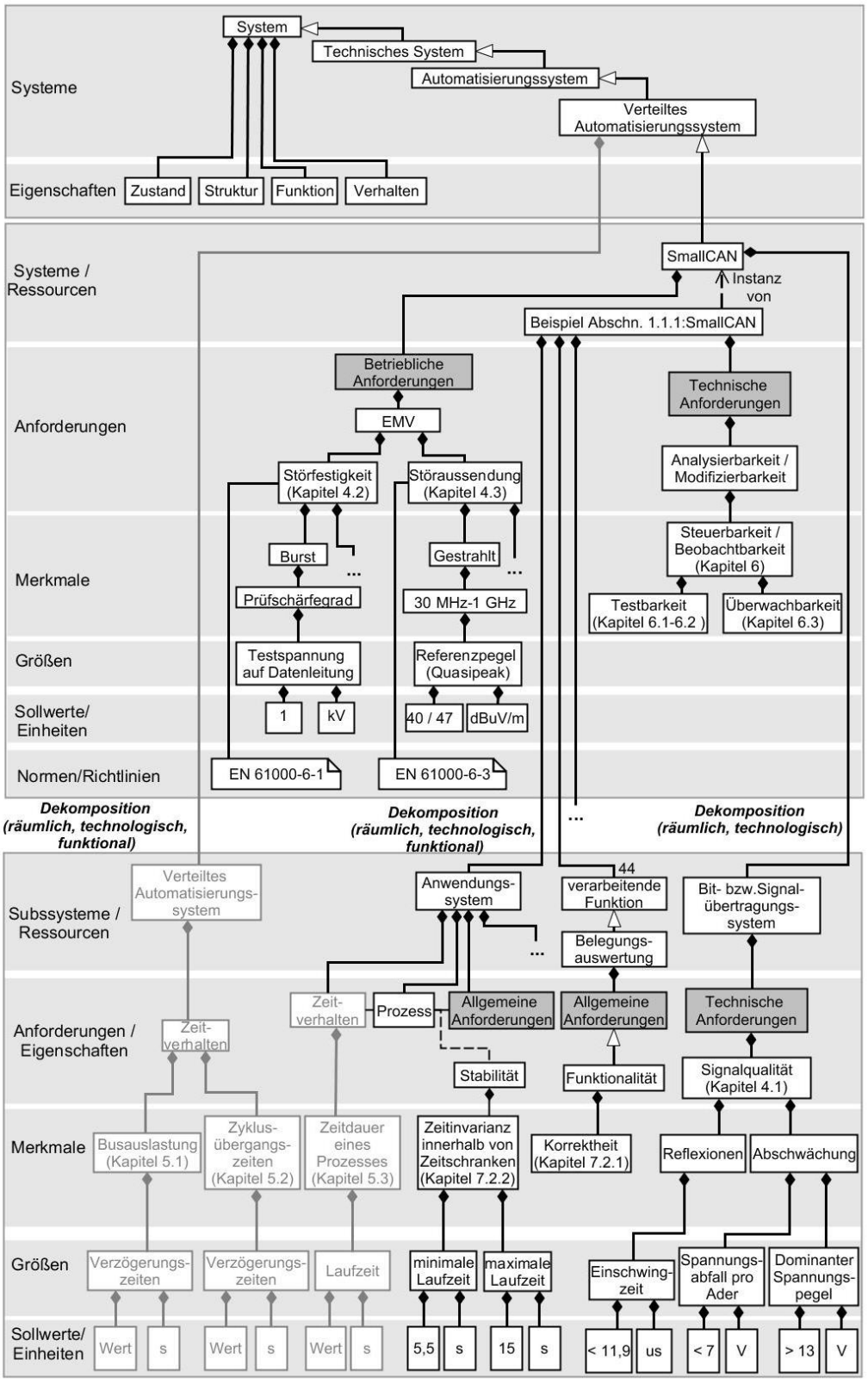


Abbildung 3.10: Ausschnitt aus dem Hierarchieschema zur Beschreibung von Anforderungen an die betrachteten Systeme

Im Kontext dieser Arbeit bezieht sich die Betrachtung temporaler Eigenschaften eines verteilten Automatisierungssystems im Wesentlichen auf die Ermittlung von Ist- und Sollwerten und nicht auf die Überprüfung zeitbedingter Anforderungen. Daher sind in Abbildung 3.10 die zu ermittelnden Anforderungen bzw. Eigenschaften in grauer Schrift dargestellt. Die *Verzögerungszeiten*, verursacht sowohl durch die Busauslastung (Abschnitt 5.1) als auch durch Zyklusübergänge zwischen Programmkomponenten (Abschnitt 5.2), werden behandelt. Bezogen auf das Anwendungsbeispiel aus Abschnitt 1.1.1 wird in Abschnitt 5.3 eine Analyse zur Erhebung von Sollwerten für die *Laufzeit* eines Steuerungsprozesses vorgestellt.

Die in Abschnitt 1.1.1 eingeführten (Beispiel-)Systeme müssen den Anforderungen der *Beobachtbarkeit* und *Steuerbarkeit* gerecht werden, damit für sie eine Online-Analyse durchgeführt werden kann. Die Erfüllung der Anforderungen an Beobachtbarkeit und Steuerbarkeit wird mittels struktureller Analysen in Kapitel 6 geprüft.

Auf Basis der getätigten Beobachtbarkeits- und Steuerbarkeitsanalysen lassen sich wiederum die Anforderungskriterien der Richtigkeit und der Laufzeit bewerten. Um den Anforderungen an *Funktionalität* bzw. *Korrektheit* gerecht zu werden, wurden exemplarisch für die Systemfunktion zur Belegungsauswertung Funktionstests durchgeführt. Das Vorgehen zur Testausführung wird in Abschnitt 7.2.1 vorgestellt. Unter Berücksichtigung der ermittelten Sollwerte für die *Zeitinvarianz* des kausalen Prozessablaufes (Abschnitt 5.3) wird in Abschnitt 7.2.2 eine Laufzeitüberwachung im Online-Betrieb vorgestellt, die ebenfalls auf Petrinetzen basiert.

3.4 Zusammenfassende Darstellung

Die in den einzelnen Schritten der Systemanalyse beschriebenen systemtheoretischen und geräteorientierten Eigenschaften sowie die formulierten Anforderungen an die jeweiligen Systeme lassen sich in einem ganzheitlichen Systemmodell zusammentragen. Die rein systemorientierte Sichtweise zur Beschreibung der zentralen Eigenschaften des Gesamtsystems (*Struktur*, *Zustand*, *Funktion* und *Verhalten*) bildet dabei die abstrakteste Ebene (*Systeme*) der Übersicht. In Anlehnung an eine konkrete Problemstellung (vgl. Abschnitt 1.2) ermöglicht die systemische (De-)Komposition eine spezifischere Betrachtung relevanter Teilsysteme (*Subsysteme*), wobei nicht zwingend nach nur einem Partitionierungsprinzip (funktional, räumlich, technologisch) vorgegangen werden muss. Eine geräteorientierte Sichtweise sieht eine ausschließliche Beschreibung von Ressourcen durch deren Qualitätseigenschaften in der Ebene der *Ressourcen* vor. Relationen zwischen Systemfunktionen und Ressourcen stellen Verknüpfungen zwischen der systemorientierten und der geräteorientierten Sichtweise

dar. Innerhalb einer vereinigten Ebene (*(Sub-)Systeme / Ressourcen*) können sowohl systemtheoretische als auch geräteorientierte Eigenschaften zusammengetragen werden. Ergänzend werden diejenigen Anforderungen an das Gesamtsystem bzw. an die dekomponierten Teilsysteme formuliert, die der jeweiligen Problemstellung zuzuordnen sind. Die *Eigenschaften* und *Anforderungen* können durch die Attributhierarchie bis auf die Ebene der *(Soll-)Werte und Einheiten* hinreichend genau formalisiert werden. Gestützt wird das Systemmodell durch die Möglichkeit zur Relationierung mit weiterführenden Dokumenten und Modellen wie beispielsweise Normen. Die Systemmodelle sind allgemein anwendbar und lediglich zur Veranschaulichung und unter Berücksichtigung der vorangegangenen Abschnitte in Abbildung 3.11 und Abbildung 3.12 vorzugsweise am Beispiel des Automatisierungssystems SmallCAN dargestellt.

In Abbildung 3.11 sind das rein funktional betrachtete *Anwendungssystem* (rot umrandet) und die konkrete *Raumautomation aus Abschnitt 1.1.1* als dessen Instanz in einem zusammenfassenden Systemmodell gezeigt. Auf systemtheoretischer Ebene wird zunächst vom allgemeinen Systembegriff bis zur Ausprägung des Systems SmallCAN konkretisiert. Darunter ist eine Ebene eingeführt, die zur vereinigten Abbildung systemtheoretischer sowie gerätetechnischer Eigenschaften des konkreten *Automatisierungssystems aus Abschnitt 1.1.1* dient. Beispielhaft ist dort die *Energieeffizienz* als Anforderung an dieses System aufgeführt. Eine Abbildung der Teilsysteme *Anwendungssystem* bzw. *Raumautomation aus Abschnitt 1.1.1* des allgemeinen *Automatisierungssystems* bzw. des konkreten *Beispielsystems aus Abschnitt 1.1.1* erfolgt in der Ebene der dekomponierten *Subsysteme* bzw. *Subsysteme / Ressourcen*. Dabei werden der Ebene *Subsysteme / Ressourcen* die systemtheoretischen und die gerätetechnischen Eigenschaften zugeordnet. Die der Raumautomation zugeordneten Systemfunktionen sind wiederum an Ressourcenkomponenten geknüpft. Beispielsweise beinhaltet die Raumautomation eine *Außenluftregelung*, die durch das Programm *AUSLG* und das Anwendungsmodul *ANMUL* realisiert wird (Anhang A). Die qualitätsbezogenen Eigenschaften dieser gerätetechnischen Komponenten (*Außenluftregler*) werden in der Ebene der *Ressourcen* formuliert.

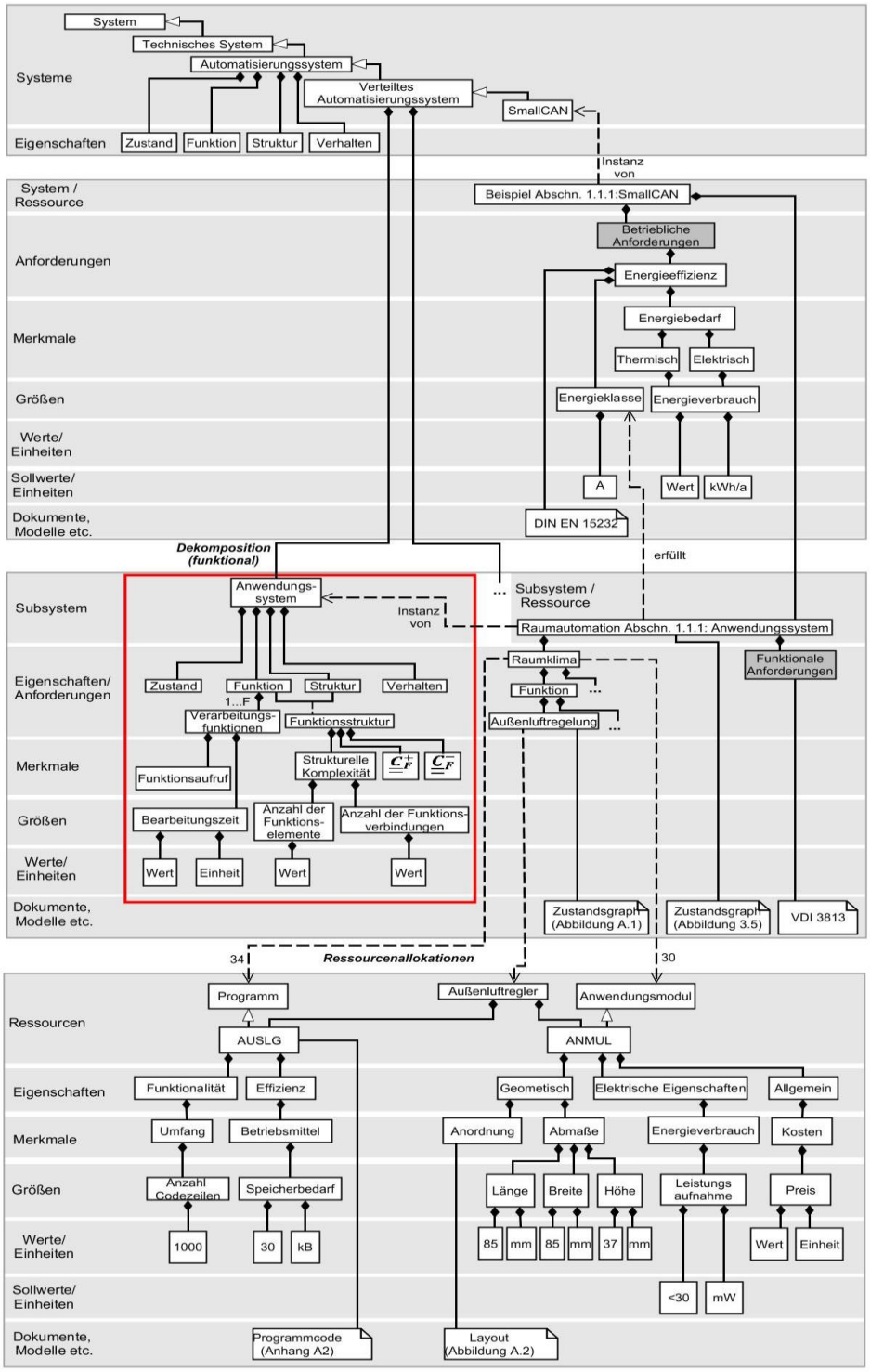


Abbildung 3.11: Ausschnitt aus einem ganzheitlichen Systemmodell mit Beschreibung eines funktional betrachteten Anwendungssystems (rot) und einer SmallCAN-Raumautomation als dessen Instanz

In Abbildung 3.12 ist ein ganzheitliches Systemmodell im Kontext des dekomponierten Datenübertragungssystems von SmallCAN dargestellt, für das eine Kombination aus den drei Partitionierungsprinzipien in Betracht gezogen wurde. In der obersten Ebene *Systeme / Ressourcen* ist exemplarisch die technisch funktionale Anforderungen der

Effizienz für das System SmallCAN angegeben. Das *Datenübertragungssystem* ist als Dekomposition des Systems *SmallCAN* skizziert und ist ausschließlich unter systemtheoretischen Aspekten, d. h. über die zentralen Systemeigenschaften in der Ebene der *Subsysteme* beschrieben. Die Systemfunktion der *Datenübertragung* wird über 2 bis *n Buskoppler*, ggf. mehreren *Busklemmen* sowie das *Busmedium* realisiert.

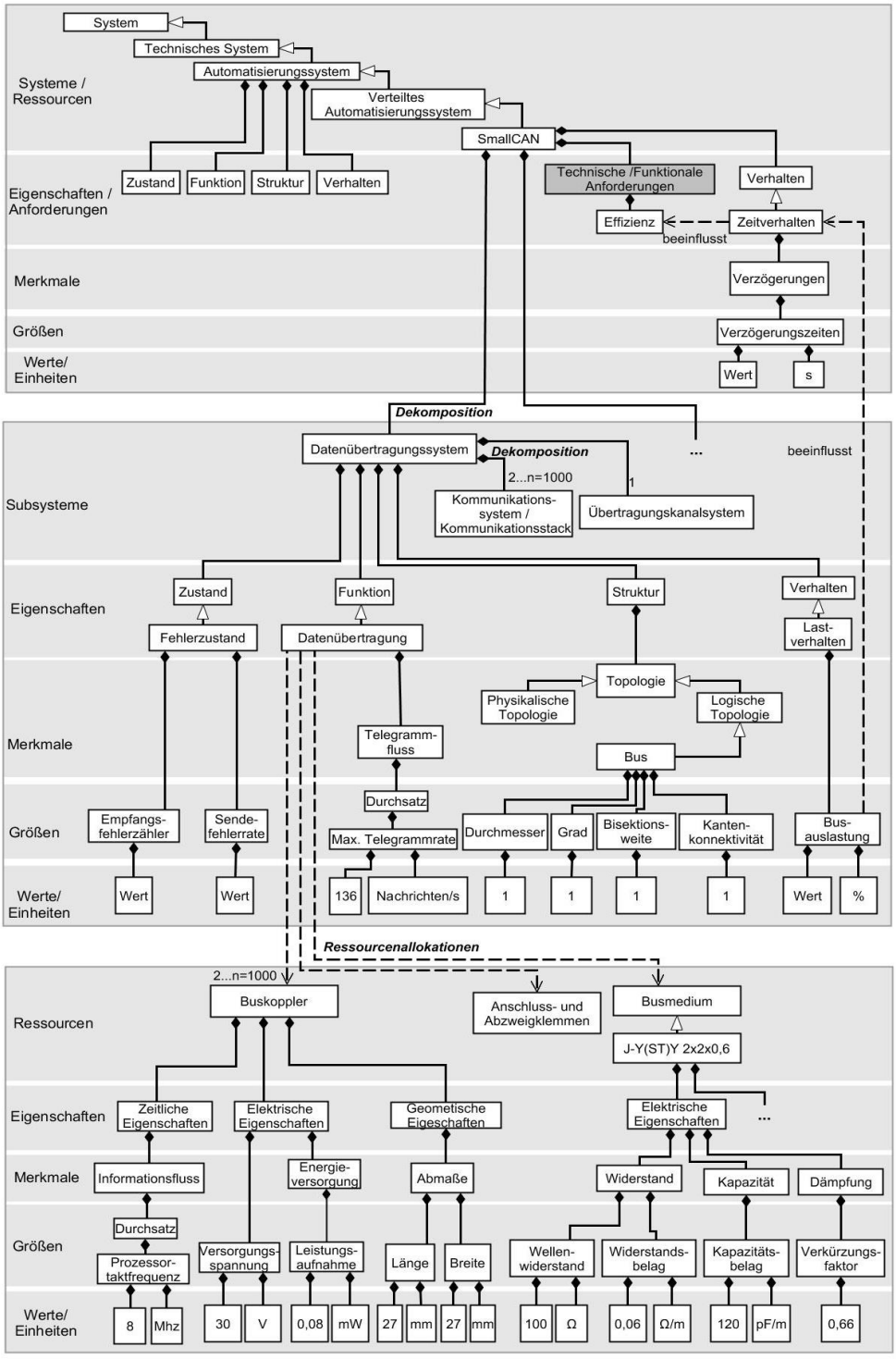


Abbildung 3.12: Ausschnitt aus dem ganzheitlichen Systemmodell zur Beschreibung des Datenübertragungssystems als Dekomposition vom System SmallCAN

4 Modellierung und Analyse der Signalqualität und der elektromagnetischen Verträglichkeit

In Kapitel 3 ist die Erstellung von Systemmodellen aufgezeigt, welche den Kontext konkreter Analyseaufgaben beschreiben, indem das jeweils notwendige Systemwissen systematisch formuliert und eindeutig zugeordnet wird. Die Systemmodelle dienen als Grundlage für weiterführende Analysen, beispielsweise für Simulationen. In diesem Kapitel werden unter Zuhilfenahme eines wissensbeschreibenden Systemmodells die Signalqualität und die elektromagnetische Verträglichkeit eines verteilten Automatisierungssystems analysiert. Der Fokus der Untersuchungen liegt auf der Simulation komplexer physikalischer Topologien eines verteilten Automatisierungssystems, d. h. räumlich stark ausgedehnte Bitübertragungssysteme.

In Abschnitt 4.1 wird zunächst das für die Analyseaufgabe heranzuziehende Systemmodell eingeführt. Die aus dem Systemmodell ableitbaren deskriptiven und analytischen Modelle, die für die simulative Untersuchung der *Signalqualität* nötig sind, werden behandelt. Die aus der Simulation sowie aus messtechnischen Analysen ermittelten Ergebnisse werden vorgestellt und diskutiert. In den Abschnitten 4.2 und 4.3 werden die durchgeführten Untersuchungen zur Störfestigkeit und die Störaussendung erörtert, entsprechende Maßnahmen unter Berücksichtigung komplexer Systemausprägungen behandelt sowie die Ergebnisse aus den Untersuchungen der *elektromagnetischen Verträglichkeit* vorgestellt.

Wesentliche Eigenschaften ausgewählter Bitübertragungssysteme sind in Tabelle 9 aufgezeigt. Im Vergleich zu den weiteren aufgeführten Lösungen zeichnet sich das System SmallCAN durch ausgedehnte Segmente mit bis zu 1000 Teilnehmern und Leitungslängen von bis zu 1000 m aus, d. h. ein maximal ausgefülltes SmallCAN-Segment ist aufgrund der weiten räumlichen Ausdehnung und der großen Anzahl von Hardwarekomponenten bei freier Topologie durch eine hohe strukturelle Komplexität charakterisiert. Die große Anzahl von Leitungsabschnitten, Bauteilkomponenten und parasitär wirkenden Elemente kann einen negativen Einfluss auf das Signalverhalten eines Segmentes im Bitübertragungssystem haben. Daher wird der Fokus der Signalqualitätsuntersuchungen aus Abschnitt 4.1 auf solche komplexen Kommunikationsarchitekturen des Systems SmallCAN gelegt. Ziel ist es, die ausgeprägten Kommunikationsarchitekturen zu modellieren und zu simulieren um die der Systemgröße geschuldeten Abweichungen im Signalverlauf zu erfassen und zu bewerten.

Tabelle 9: Merkmale verschiedener Bitübertragungssysteme

Merkmale/ Kenngrößen	**Bussystem/Physikalische Schicht**		
	SmallCAN	**CAN**	**RS485 (z. B. LON)**
Max. Anzahl an Teilnehmern	1000	128	64
Max. Leitungslänge (Knoten zu Knoten)	1000 m	500 m	500 m
Topologie	Frei	Linie empfohlen	Linie empfohlen
Signalübertragung	Unsymmetrisch	Symmetrisch (um 2,5 V)	Symmetrisch (um 0 V)
Spannungsdifferenz	Rezessiv: 23 V	Dominant: > 2 V	10 V
Terminierung	Nein	Ja	Ja
Einspeisung	Zentral	Teilnehmer	Teilnehmer
Transceiver	Diskret	Integriert	Integriert
Buskabeltyp	Telefonleitung J-Y(St)Y 2x2 x0,6 mm	Buskabel nach ISO 11898−2 (2x2x0,8 mm)	Belden 85102 (für max. Leitungslänge)

Die für SmallCAN genutzte unsymmetrische Signalübertragung mit zentraler Datenstromeinspeisung stellt die einfachste und kostengünstigste Art der Datenübertragung dar. Die Signalspannung auf der Datenleitung ändert sich dabei gegenüber einem Bezugspotenzial nahe der elektrischen Masse, die über eine Leitungsader mitgeführt wird. Bei einer symmetrischen Signalübertragung (CAN, RS485) ist eine Spannungssymmetrie zum Bezugspotenzial zu erzeugen, wodurch Mehrkosten für die Elektronik, beispielsweise durch die Verwendung eines Subtrahierers, in jedem Buskoppler zu berücksichtigen sind. Weiterhin entsteht ein Mehraufwand für die Verkabelung (verdrillte Aderpaare mit Hin- und Rückleiter, zusätzliche getrennte Masseverkabelung zur Verringerung des Störeinflusses durch Masseschleifen). Für eine kostenneutrale Realisierung stark verteilter Übertragungssysteme ist jedoch der Aufwand für die Installation gering zu halten, so

dass in diesem Fall die Verwendung einer unsymmetrischen Signalübertragung eine bessere Wahl darstellt.
Die unsymmetrische Signalübertragung über schwach verdrilltes Installationskabel ohne aufgelegten Schirm vereinfacht zwar den Verkabelungsaufwand, ist aber deutlich anfälliger gegenüber elektromagnetischen Einflüssen. Beispielsweise können eingekoppelte Gleichtaktstörsignale (*Common Mode*) zu unterschiedlich großen Strömen auf den unsymmetrischen Daten- und Masseleitung führen. Die an den ungleichen Impedanzen anliegenden Störspannungen können so zu einer Gleich/Gegentakt-Konversion und damit zu einer deutlichen Verschlechterung der Störfestigkeit (Abschnitt 4.2) führen, die in einem direkten Zusammenhang mit der Signalqualität steht (vgl. Abbildung 4.1).
Im Gegensatz zu den Standards CAN und RS485 ist die Hardware eines SmallCAN-Transceivers diskret aufgebaut und als Busmedium kommt ein einfaches Telefonkabel mit Sternverseilung zum Einsatz (vgl. Abbildung 3.9). D. h. dem vernetzten Betrieb vieler räumlich verteilter und einfach aufgebauter Busteilnehmer über ein einfaches schwach verdrilltes Installationskabel ohne aufgelegten Schirm ist eine besondere Aufmerksamkeit hinsichtlich der *Störaussendung* zu widmen. Daher werden in Abschnitt 4.3 die durchgeführten Untersuchungen zur Reduzierung der elektromagnetischen Störausstrahlung erörtert sowie Maßnahmen und Ergebnisse vorgestellt.
Der nur schwer zu vergleichende physikalische Aufbau und die entsprechend abweichende Konzeptionierung von SmallCAN gegenüber den standardisierten Lösungen verdeutlicht die erforderlichen Analysen zur Bewertung der Signalqualität und der elektromagnetischen Verträglichkeit für dieses System.

4.1 Signalqualität

Die Untersuchungen des Bitübertragungssystems von SmallCAN hinsichtlich des Signalverhaltens erfolgten im Wesentlichen durch Simulationen [Diekhake 2011], [Diekhake 2013]. Eine umfassende messtechnische Absicherung ist ohne einen deutlichen Mehraufwand nicht möglich, da für das Bitübertragungssystem viele Parameter, beispielsweise die Anzahl der Teilnehmer, die physikalische Topologie oder Charakteristika der Bauelemente zu berücksichtigen sind. Es ergeben sich unüberschaubar viele Einflussfaktoren, wobei die Ermittlung sämtlicher Auswirkungen auf das Signalverhalten selbst mit simulativen Methoden eines hohen Aufwandes bedarf. Für die Modellierung und Simulation der elektrischen Schaltungen eignet sich insbesondere die Softwareumgebung SPICE [Guezgouz et al. 2010], [Zdenek/Jiri 2013], [Diekhake 2013]. Sie bietet die Basis für eine textbasierte

Beschreibung von Bauelementen, Schaltkreisen und deren Verknüpfungen. SPICE-basierte Werkzeuge (Tabelle 7 in Abschnitt 2.2.2) stellen außerdem die Grundfunktionen zum algorithmischen Auffinden von Näherungslösungen für systembeschreibende Differentialgleichungen zur Verfügung, die die Basis für eine zeit- und wertekontinuierliche Simulation bilden.

In den folgenden drei Abschnitten werden die Modellierungsansätze zur Analyse der Signalqualität vorgestellt. Zunächst wurde ein Systemmodell erstellt, welches das Bitübertragungssystem über seine *Eigenschaften*, *Merkmale*, *Größen*, *Werte* sowie *Einheiten* beschreibt (Abschnitt 4.1.1). Aus dem Systemmodell (vgl. Kapitel 3) lässt sich ein deskriptives Modell ableiten, welches die Struktur des Bitübertragungssystems sowie die Strukturkopplung zur Umwelt umfasst (Abschnitt 4.1.2). Weiterhin stellt das Systemmodell die Informationsquelle zur Parametrierung mathematisch analytischer Modelle (Abschnitt 4.1.3) dar, auf deren Basis die Simulationen erfolgen. Die durchgeführte Validierung eines Simulationsmodells wird in Abschnitt 4.1.4 behandelt. Weiterhin werden ausgewählte Simulationsergebnisse vorgestellt, die die Funktionsfähigkeit des Bitübertragungssystems für maximal ausgeprägte physikalische Topologien bestätigen. Dafür wurden die Beeinflussungen durch eine erhöhte Teilnehmeranzahl (Abschnitt 4.1.5) durch die räumliche Ausdehnung des Netzwerkes (Abschnitt 4.1.6) und durch verschiedene Varianten des physikalischen Aufbaus (Abschnitt 4.1.7) simulativ untersucht.

4.1.1 Systemmodell

Das der Analyseaufgabe entsprechende Systemmodell kann Abbildung 4.1 entnommen werden. In Anlehnung an Abbildung 3.6 stellt das Systemmodell das ganzheitliche *Bitübertragungssystem* (Dekomposition vom System SmallCAN) und die nötigen Ressourcen (*Busmedium*, *Anschluss- und Abzweigeklemmen*, *Datenstromeinspeisung*, *Transceiver*) zur Realisierung der Systemfunktion der *Signalübertragung* dar. Gemäß den Anforderungen aus Abbildung 3.10 sind für die Signalqualitätsanalyse lediglich das Bitübertragungssystem und für die elektromagnetische Verträglichkeit das Gesamtsystem zu bewerten.

Die Struktur des Bitübertragungssystems wird über die *physikalische Topologie* bzw. in Adjazenz-Matrixdarstellung über $\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{phy}}$ formuliert. Analog der *Strukturkopplung zur Umwelt* nach Abbildung 3.2 ist eine *Pegel-Instanziierung* [Schrom 2003: 111] abgebildet. Die Anforderung an eine ausreichende Signalqualität (*Technische Anforderung*) kann u. a. durch die Angabe der Verhaltensmerkmale *Reflexion* und *Abschwächung* näher beschrieben werden, denen wiederum z. B. die Kenngrößen *Einschwingzeit* und *Spannungsabfall* zugeordnet sind. Für alle topologischen

Ausprägungen ist zu garantieren, dass die in der Ebene der *Sollwerte / Einheiten* hinterlegten Grenzwerte für die *Reflexion* und die *Abschwächung* sowie für den *Leitungszustand* nicht über bzw. unterschritten werden. Die Beschreibung der Ressourcen erfolgt im Kontext der Problemstellung im Wesentlichen durch die Angabe elektrischer Eigenschaften. Als *Busmedium* wird ein einfaches Telefonkabel gemäß Abbildung 3.9 verwendet. Weiterhin sind *Anschluss- und Abzweigeklemmen* für das Durchschleifen oder Verzweigen der Busleitung notwendig. Eine *Datenstromeinspeisung* ist beschrieben, die im Falle eines rezessiven Signalpegels eine *Quellenspannung* von 24 V bereitstellt und im Falle eines dominanten Signalpegels einen *Quellenstrom* von 230 mA in die Datenleitung einspeist. Die Erzeugung des dominanten, physikalischen Pegels erfolgt aus einem Umgebungs- bzw. Stimulationsmodell heraus über eine Instanziierung eines logischen Pegels (*Strukturkopplung mit der Umwelt*) und der anschließenden Stimulation eines Schalttransistors innerhalb des zu modellierenden Bitübertragungssystems, wodurch die Datenleitung auf das Bezugspotenzial nahe GND gelegt wird. Bezugnehmend auf die Untersuchungen zur elektromagnetischen Verträglichkeit (Abschnitt 4.2 und Abschnitt 4.3) sind ergänzend auf der Ebene *Systeme / Ressourcen* die betrieblichen Anforderungen der *Störaussendung* und der *Störfestigkeit* auszugsweise konkretisiert. Exemplarisch ist der *Referenzpegel* als Grenzwert für die gestrahlte Störaussendung quantifiziert. Für die Störfestigkeit ist als zu erfüllendes Anforderungsattribut die bei der Burst-Prüfung aufgeprägte *Testspannung auf den Datenleitungen* angegeben.

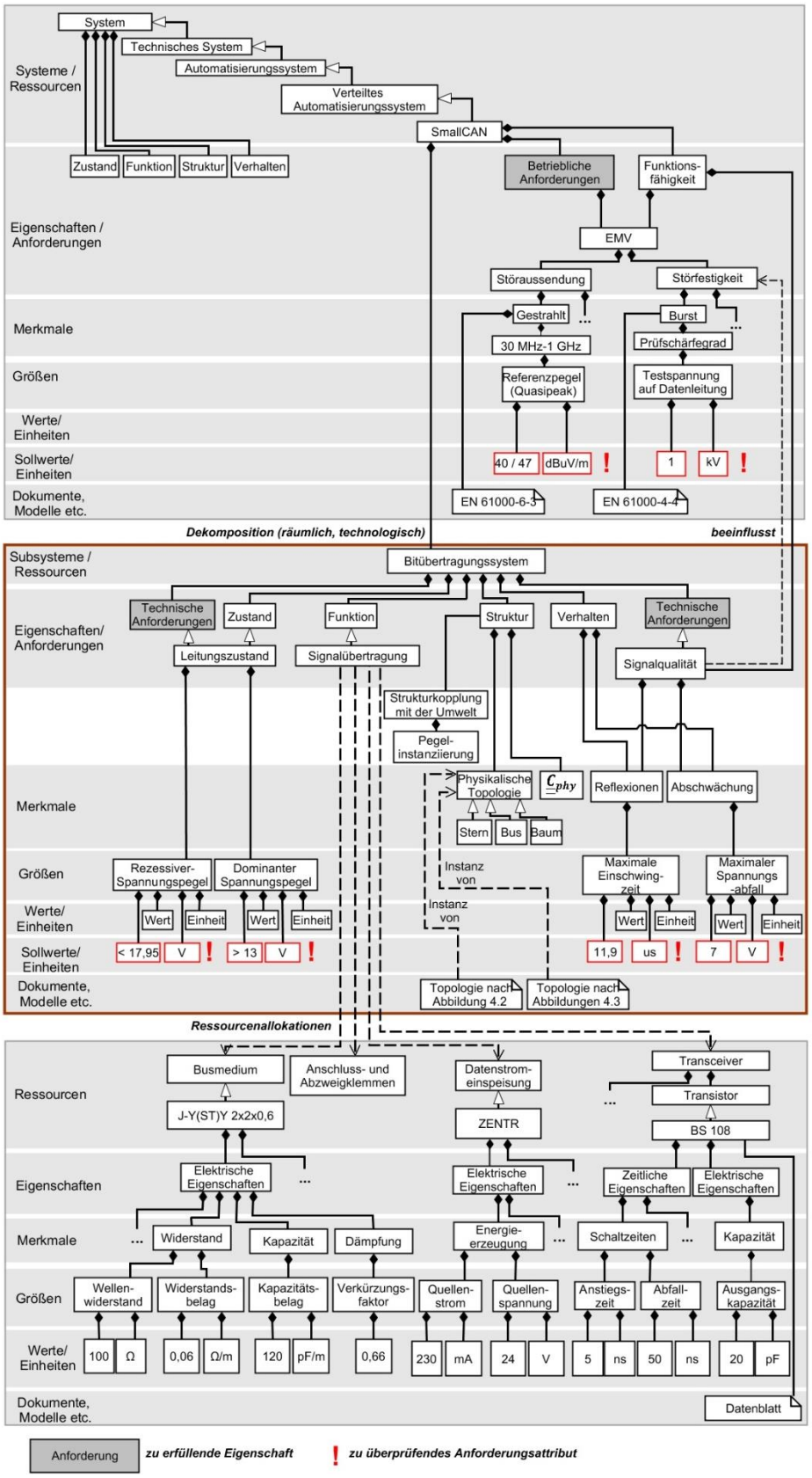


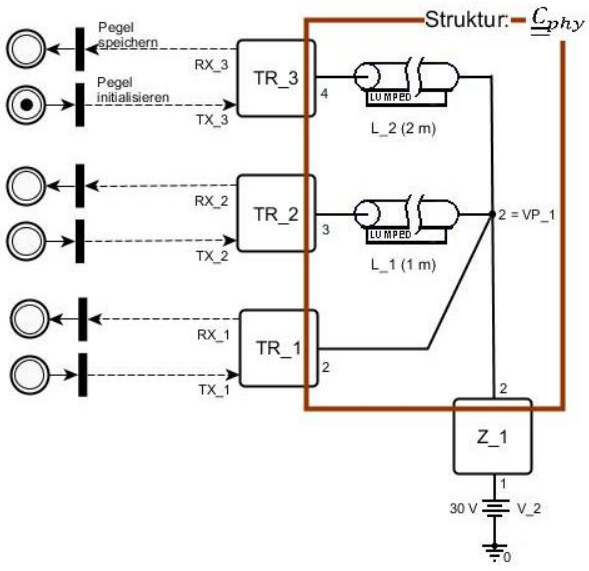
Abbildung 4.1: Ganzheitliches Systemmodell, angepasst an die SmallCAN-spezifischen Problemstellungen zur elektromagnetischen Verträglichkeit und zur Signalqualität, mit räumlich technologisch betrachteten Bitübertragungssystem (braun) als Dekomposition von SmallCAN

4.1.2 Deskriptive Modelle

Der physikalische Aufbau des Bitübertragungssystems, d. h. die physikalische Topologie, kann in Adjazenz-Matrixdarstellung formuliert werden, aus der übergeordnete SPICE-Netzlisten für eine Simulations-Umgebung generiert werden können. Eine Umsetzung zur automatisierten Erstellung der Netzlisten auf Basis von Matrixdarstellungen wurde beispielsweise in [Chu 2014] erarbeitet.
Für eine exemplarische minimale Topologie des Bitübertragungssystems mit der symmetrischen Adjazenzmatrix

$$\underline{\underline{C}}_{phy} = \begin{array}{c} \text{Knoten} \\ \begin{matrix} TR\_1=2 & TR\_2=3 & TR\_3=4 & Z\_1=2 & VP\_1=2 \end{matrix} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & \text{L\_1} \\ 0 & 0 & 0 & 0 & \text{L\_2} \\ 0 & 0 & 0 & 0 & 1 \\ 1 & \text{L\_1} & \text{L\_2} & 1 & 0 \end{pmatrix} \end{array} \begin{array}{l} TR\_1=2 \\ TR\_2=3 \\ TR\_3=4 \\ Z\_1=2 \\ VP\_1=2 \end{array} \quad \text{Knoten}$$

zeigt Abbildung 4.2 die Struktur aus drei Transceivern (TR_1 bis TR_3), einer Datenstromeinspeisung (Z_1) und einem Verbindungspunkt (VP_1) sowie das zugehörige SPICE Netzlistenformat.



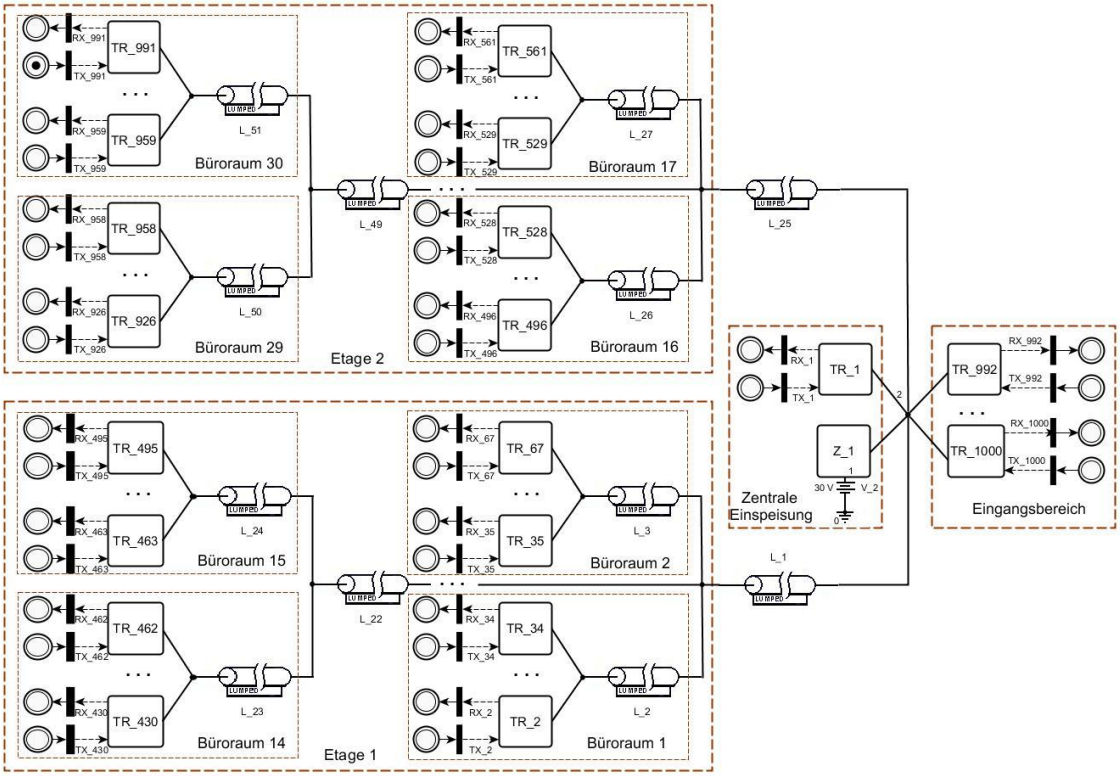
V_2	1	0			30Vdc
Z_1	0	2	1		DATA_SUPPLY
TR_1	TX_1	2	0	RX_1	BUS_TRANSCEIVER
L_1	3	2	GND_1		TLUMP (LEN=1 R'=0.06 C'=120p)
TR_2	TX_2	3	GND_1	RX_2	BUS_TRANSCEIVER
L_2	4	2	GND_2		TLUMP (LEN=2 R'=0.06 C'=120p)
TR_3	TX_3	4	GND_2	RX_3	BUS_TRANSCEIVER

Abbildung 4.2: Deskriptives Modell eines minimalen Aufbaus und beschreibende Netzliste in SPICE

Zwei Busleitungen von je 1 m und 2 m verbinden die Datenstromeinspeisung Z_1 und den Transceiver TR_1 mit den Transceivern TR_2 und TR_3 in Form eines physikalischen Sterns mit dem Sternpunkt VP_1. An die logischen Signalein- und ausgänge TX und RX der Transceiver kann über die Koppelelemente des Modells (in Abbildung 4.2 als Petrinetz-Notation dargestellt) ein Umgebungs- bzw. Stimulationsmodell angebunden werden.

Abbildung 4.3 (oben) zeigt exemplarisch eine maximale physikalische Struktur des Bitübertragungssystems in einer Baumtopologie mit 1000 Transceivern, 46 Verbindungspunkten und einer Datenstromeinspeisung bei einer Gesamtleitungslänge von 1000 m. Die Topologie entspricht einer vereinfachten Abbildung einer typischen Struktur eines Bürogebäudes mit Eingangsbereich (9 Gerätekomponenten) sowie zwei Etagen mit je 15 Büroräumen, in denen je 33 Gerätekomponenten für eine Einzelraumregelung sternförmig angeordnet sind. D. h. von der zentral angeordneten Datenstromeinspeisung führen zwei Hauptstränge ab, die alle 20 m je zwei ca. 20 m lange Stichleitungen aufweisen. An den Stichleitungen ist je ein Netzwerk (Büroraum) aus 33 verbundenen Transceivern angeschlossen. Die zugehörige Netzliste ist in Abbildung 4.3 (unten) auszugsweise abgebildet.

Zur Durchführung einer realitätsnahen Simulation eines Bitübertragungssystems wurden neben den Topologiestrukturen auch die inneren Strukturen der drei Ressourcenkomponenten *Transceiver, Datenstromeinspeisung* und *Busmedium* über SPICE-Modelle beschrieben. Für diese wurden ebenfalls Netzlisten erstellt. Die wesentlichen zu berücksichtigenden Parameterwerte der Komponentenmodelle können dem Bitübertragungssystem zugeordneten Ressourcenbeschreibungen aus Abbildung 4.1 entnommen werden.
Ein Transceiver wird durch das Schaltungsmodell BUS_TRANSCEIVER repräsentiert, welches den physikalischen Ausgang zum Busmedium, den Masseanschluss GND, den logischen Ausgang der empfangenen Daten sowie den logischen Eingang der zu versendenden Daten anbietet. Ein dominanter Signalpegel, der über ein Umgebungsmodell instanziiert wird, wird durch das Schalten eines Transistors erzeugt, wodurch der Signalpegel am physikalischen Ausgang in Richtung GND gezogen wird. Die Datenstromeinspeisung DATA_SUPPLY stellt den rezessiven Signalpegel sowie die Konstantstromquelle bereit. Das Modell verfügt über drei Übergabestellen für die Masse, die Eingangsstromversorgung und den Ausgang zum Busmedium. Für das Modell des Busmediums kann u. a. die in SPICE bereits vorhandene Beschreibung TLUMP verwendet werden.



V_2	1	0			30Vdc
Z_1	0	2	1		DATA_SUPPLY
TR_1	TX_1	2	0	RX_1	BUS_TRANSCEIVER
L_1	3	2	GND_1		TLUMP (LEN=20 R'=0.06 C'=120p)
L_2	4	3	GND_2		TLUMP (LEN=20 R'=0.06 C'=120p)
TR_2	TX_2	4	GND_2	RX_2	BUS_TRANSCEIVER
...					
TR_34	TX_34	4	GND_2	RX_24	BUS_TRANSCEIVER
L_3	5	3	GND_3		TLUMP (LEN=20 R'=0.06 C'=120p)
TR_35	TX_35	5	GND_3	RX_35	BUS_TRANSCEIVER
...					
TR_67	TX_67	5	GND_3	RX_67	BUS_TRANSCEIVER
...					
L_22	23	22	GND_22		TLUMP (LEN=20 R'=0.06 C'=120p)
L_23	24	23	GND_23		TLUMP (LEN=20 R'=0.06 C'=120p)
TR_430	TX_430	24	GND_23	RX_430	BUS_TRANSCEIVER
...					
TR_462	TX_462	24	GND_23	RX_462	BUS_TRANSCEIVER
L_24	25	23	GND_24		TLUMP (LEN=20 R'=0.06 C'=120p)
TR_463	TX_463	25	GND_24	RX_463	BUS_TRANSCEIVER
...					
TR_495	TX_495	25	GND_24	RX_495	BUS_TRANSCEIVER
L_25	25	2	GND_25		TLUMP (LEN=20 R'=0.06 C'=120p)
L_26	26	25	GND_26		TLUMP (LEN=20 R'=0.06 C'=120p)
TR_496	TX_496	26	GND_26	RX_496	BUS_TRANSCEIVER
...					
TR_528	TX_528	26	GND_26	RX_528	BUS_TRANSCEIVER
L_27	27	25	GND_27		TLUMP (LEN=20 R'=0.06 C'=120p)
TR_529	TX_529	27	GND_27	RX_529	BUS_TRANSCEIVER
...					
TR_561	TX_561	27	GND_27	RX_561	BUS_TRANSCEIVER
...					
L_49	46	2	GND_46		TLUMP (LEN=20 R'=0.06 C'=120p)
L_50	47	46	GND_47		TLUMP (LEN=20 R'=0.06 C'=120p)
TR_926	TX_926	47	GND_47	RX_926	BUS_TRANSCEIVER
...					
TR_958	TX_958	47	GND_47	RX_958	BUS_TRANSCEIVER
L_51	48	46	GND_48		TLUMP (LEN=20 R'=0.06 C'=120p)
TR_959	TX_959	48	GND_48	RX_959	BUS_TRANSCEIVER
...					
TR_991	TX_991	48	GND_48	RX_991	BUS_TRANSCEIVER
TR_992	TX_992	2	0	RX_992	BUS_TRANSCEIVER
...					
TR_1000	TX_1000	2	0	RX_1000	BUS_TRANSCEIVER

Abbildung 4.3: Deskriptives Modell eines maximalen Aufbaus (oben) und beschreibende Netzliste in SPICE (unten)

4.1.3 Analytische Modelle

Die Komponentenmodelle werden zur Erzielung einer einfachen Berechenbarkeit durch Ersatzschaltbilder dargestellt. Die einfachste Betrachtung eines Bitübertragungssystems besteht aus einer Reihenschaltung von Widerstand, Induktivität und Kapazität zur Beschreibung des Busmediums (der Leitwert G wird vernachlässigt), einer stationären Spannungsquelle mit $U_0 = 24\,\mathrm{V}$ als vereinfachtes Modell der Datenstromeinspeisung und einem einfachen Schalter als Repräsentant des Schalttransistors innerhalb des Transceivers. Das Bitübertragungssystem kann in dieser Form als ein Reihenschwingkreis gemäß Abbildung 4.4 dargestellt werden. Die folgenden Differentialgleichungen beschreiben diesen Schwingkreis.

$$\frac{\mathrm{d}^2U}{\mathrm{d}t^2} + \frac{R}{L}\frac{\mathrm{d}U}{\mathrm{d}t} + \frac{1}{CL}U = \frac{U_0}{CL} \quad \text{, wenn der Schalter geöffnet ist.} \tag{4.1}$$

$$\frac{\mathrm{d}^2U}{\mathrm{d}t^2} + \frac{R}{L}\frac{\mathrm{d}U}{\mathrm{d}t} + \frac{1}{CL}U = 0 \quad \text{, wenn der Schalter geschlossen ist.} \tag{4.2}$$

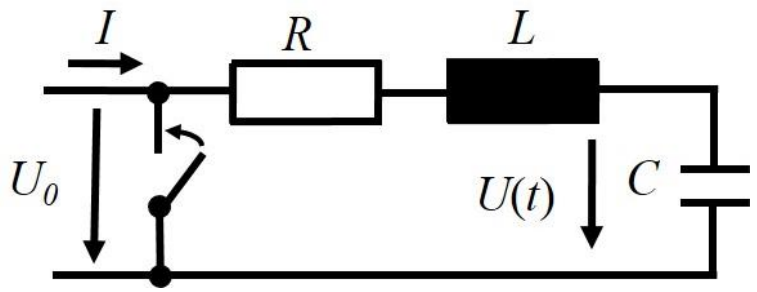


Abbildung 4.4: Vereinfachte Betrachtung des Bitübertragungssystems mittels Reihenschwingkreis

Abbildung 4.5 zeigt die Lösungen der Differentialgleichung (4.1) bzw. (4.2) für verschiedene Leitungslängen (Minimum: 1 m, Maximum: 1000 m). Dazu wurden gemäß den in Abbildung 4.1 hinterlegten Leitungsbelägen des Busmediums die folgenden Kennwerte entsprechend der jeweiligen Leitungslänge gewählt:

$$C = \begin{cases} 120\mathrm{pF}, \text{wenn } l = 1\,\mathrm{m} \\ 120\mathrm{nF}, \text{wenn } l = 1000\,\mathrm{m} \end{cases}$$

$$R = \begin{cases} 60\,\mathrm{m\Omega}, \text{wenn } l = 1\,\mathrm{m} \\ 60\,\Omega\,, \text{wenn } l = 1000\,\mathrm{m} \end{cases}$$

$$L = \begin{cases} 75\mathrm{nH}, \text{wenn } l = 1\,\mathrm{m} \\ 75\mu\mathrm{H}, \text{wenn } l = 1000\,\mathrm{m} \end{cases}.$$

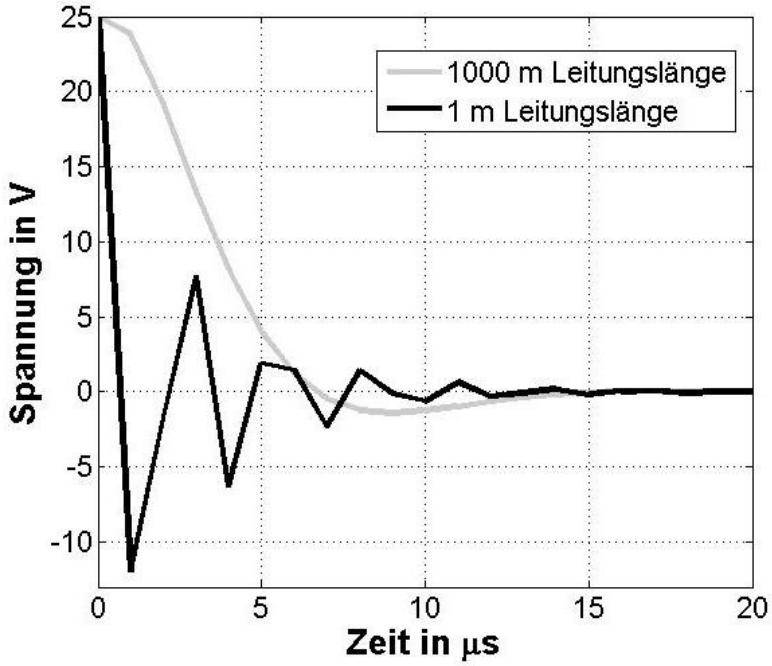


Abbildung 4.5: Signalverhalten nach dem Schalten des Reihenschwingkreises für verschiedene Parameterwerte

Das stark vereinfachte Modell ist u. a. dahingehend eingeschränkt, dass es das frequenzabhängige Verhalten des Busmediums nur dann korrekt abbilden kann, wenn die Verzögerung durch die Signallaufzeit über dem betrachteten Leitungssegment geringer ist als die Flankensteilheit des zu übertragenden Signals. Diese Problematik lässt sich jedoch lösen, indem die Leitung in eine Anordnung mehrerer Teilleitungen mit geringen Signallaufzeiten zerlegt wird. Das Leitungssegment besteht nunmehr aus einer endlichen Menge in Reihe angeordneter Vierpole gemäß Abbildung 4.6. Die Differentialgleichung für den zeitlichen Verlauf der Spannung auf der Leitung mit i Vierpolen kann dann wie folgt angegeben werden:

$$\frac{\mathrm{d}^2 U_i}{\mathrm{d}t^2} - \frac{1}{LC}(U_{i-1} - 2U_i + U_{i+1}) - \frac{R}{L}\frac{\mathrm{d}U_i}{\mathrm{d}t} = 0$$

mit $C = C_{i-1} = C_i = C_{i+1}$

$$R = R_{i-1} = R_i = R_{i+1}$$

$$L = L_{i-1} = L_i = L_{i+1}. \tag{4.3}$$

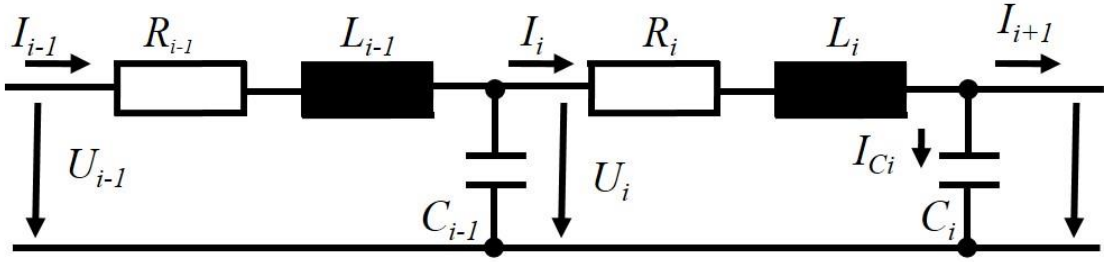


Abbildung 4.6: Beschreibung der Leitung über periodisch angeordnete Vierpole diskreter Bauelemente

Die Methode der Zerlegung einer Übertragungsleitung in endlich viele Teilleitungen mit diskreten Bauteilen wird standardmäßig für die Modellierung von Leitungen in SPICE verwendet und ist in dem Modell TLUMP umgesetzt. Insbesondere für ausgeprägte Topologien sind jedoch entsprechend viele solcher Vierpole zur Abbildung des gesamten Busmediums zu verschalten, so dass die Ermittlung der Lösung der Differentialgleichung nach (4.3) aufgrund eines erhöhten Rechenaufwandes sehr zeitintensiv ausfallen kann.

Der Ansatz der linearen Verkettung der Vierpole zu einem Leitungssegment ist ein Näherungsansatz zur Lösung der Telegrafengleichung aus (4.4), die den homogenen Spannungsverlauf auf der Leitung nach Abbildung 4.7 beschreibt:

$$\frac{\mathrm{d}^2U(x,t)}{\mathrm{d}x^2} = L'C'\frac{\mathrm{d}^2U(x,t)}{\mathrm{d}t^2} + R'C'\frac{\mathrm{d}U(x,t)}{\mathrm{d}t} \tag{4.4}$$

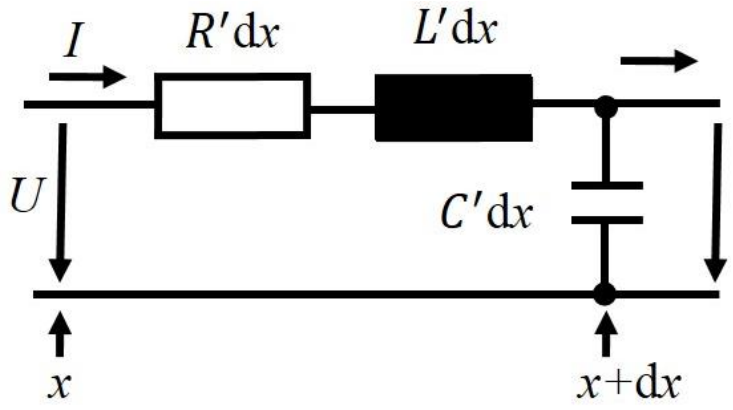


Abbildung 4.7: Beschreibung der Leitung über homogen verlaufende Bauteilkomponenten

Die allgemeine Lösung der partiellen Differentialgleichung (4.4) ist die Superposition zweier Funktionen in Abhängigkeit von Raum und Zeit in positive und negative *x*-Richtung:

$$U(x,t) = U_{\rightarrow}(x,t) + U_{\leftarrow}(x,t) = U_1\,\mathrm{e}^{\mathrm{j}\omega t-\gamma t} + U_2\,\mathrm{e}^{\mathrm{j}\omega t+\gamma t} \tag{4.5}$$

Die allgemeine Lösung (4.5) wurde u. a. in [McCammon 2011b] schaltungstechnisch abgebildet. Die entsprechende Implementierung in Form eines Leitungsmodells für SPICE wird in [McCammon 2011a] vorgestellt. Für das Bitübertragungssystem aus Abbildung 4.8 konnte durch die Anwendung dieser Implementierung die Simulationszeit um das Zehnfache gegenüber der Näherungslösung mittels diskreter Vierpolzerlegung reduziert werden.

Bei der Betrachtung des Reihenschwingkreises aus Abbildung 4.4, als ein vereinfachtes Modell des Bitübertragungssystems ist weiterhin zu berücksichtigen, dass die stationäre Spannungsquelle im Falle eines Kurzschlusses einen Strom größer 230 mA einspeisen würde und dass der dargestellte Schalter nicht das reale Schaltverhalten des Ausgangstransistors abbildet. Entsprechend sind diese stark vereinfachten Bauteile ebenfalls durch geeignetere Modelle zu ersetzen. Zur realitätsnahen Abbildung der Datenstromeinspeisung und des Bustransceivers sind daher die Modelle DATA_SUPPLY und BUS_TRANSCEIVER zu verwenden, die in [Diekhake 2013] vorgestellt wurden.

4.1.4 Validierung

Nach der Abbildung der Struktur des Bitübertragungssystems mit deskriptiven Modellen (Abschnitt 4.1.2) und der Darstellung der Ressourcenkomponenten mittels analytischer Modelle (Abschnitt 4.1.3) ließ sich das Bitübertragungssystem in einer SPICE-Umgebung simulieren. Zur Bewertung der Simulationsergebnisse erfolgte zunächst eine Modellvalidierung, indem die erhobenen Messergebnisse eines real implementierten Systems mit dem Signalverhalten aus der Simulation verglichen wurden. Der in Abbildung 4.8 exemplarisch gezeigte Aufbau gemäß

$$\underline{\underline{C}}_{phy} = \begin{array}{c} \text{Knoten} \\ \begin{array}{cccccccc} TR\_1=2 & TR\_2=3 & TR\_3=5 & TR\_4=6 & TR\_5=7 & Z\_1=2 & VP\_1=2 & VP\_2=4 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \text{L\_1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \text{L\_3} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \text{L\_4} \\ 0 & 0 & 0 & 0 & 0 & 0 & \text{L\_5} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & \text{L\_1} & 0 & 0 & \text{L\_5} & 1 & 0 & \text{L\_2} \\ 0 & 0 & \text{L\_3} & \text{L\_4} & 0 & 0 & \text{L\_2} & 0 \end{pmatrix} \end{array} \begin{array}{l} TR\_1=2 \\ TR\_2=3 \\ TR\_3=5 \\ TR\_4=6 \\ TR\_5=7 \\ Z\_1=2 \\ VP\_1=2 \\ VP\_2=4 \end{array} \text{ Knoten}$$

besteht aus fünf Busteilnehmern, die über ein Übertragungskanalsystem mit einer Gesamtleitungslänge von 900 m und zwei Verbindungspunkten verknüpft sind. Die

Spannungsverläufe aus Abbildung 4.9 wurden nah am aktiven Transceiver TR_3 an Leitungspunkt 5, an TR_4 in 400 m Entfernung vom aktiven Transceiver (Leitungspunkt 6) und nah an der Datenstromeinspeisung (Leitungspunkt 2) simuliert sowie messtechnisch erfasst. Im Folgenden werden die Ergebnisse aus der Simulation und der Messung unter Berücksichtigung analytischer Berechnungen verglichen und diskutiert.

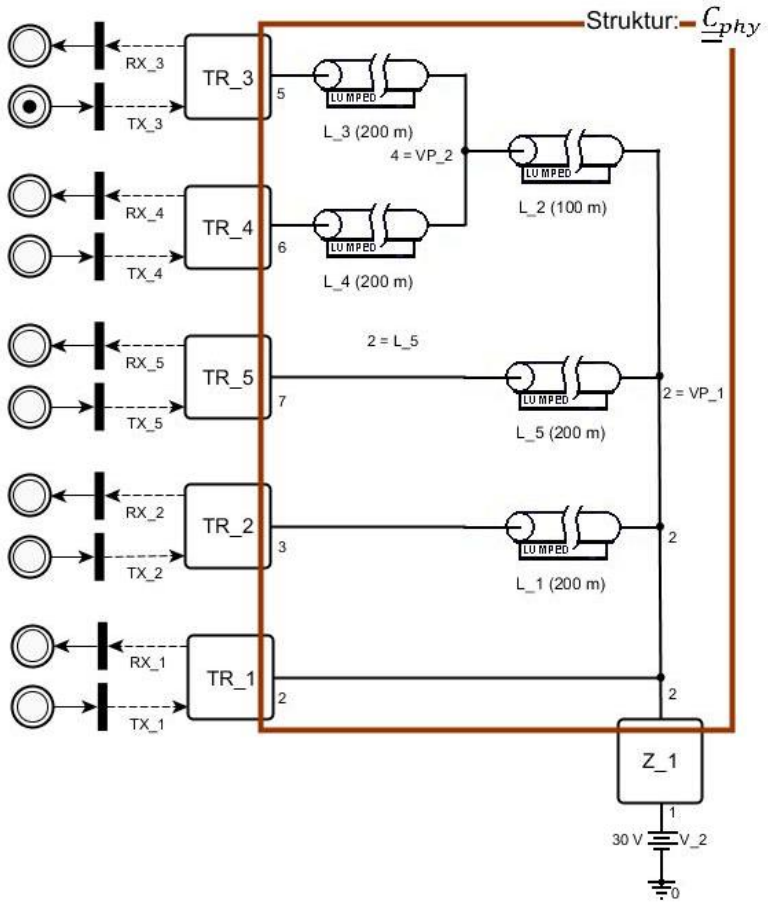


Abbildung 4.8: Deskriptives Modell eines zu validierenden Aufbaus mit 5 Transceivern und einer Gesamtleitungslänge von 900 m

Spannungsabfall

Bei einem dominanten Signalpegel führt der Stromfluss durch die Datenleitung zu einem Spannungsabfall. Wie in Abbildung 4.9 gezeigt, beträgt dieser im stationären Zustand zwischen TR_3 und TR_4 etwa 3,6 V und deckt sich mit dem über einen ohmschen Widerstand berechenbaren Spannungsabfall:

$$U = R \cdot I = \rho \frac{l}{A} \cdot I \tag{4.6}$$

Für das verwendete Busmedium J-Y(ST)Y 2x2x0,6 mm mit $\rho = 1{,}67 \cdot 10^{-2} \frac{\Omega\,\mathrm{mm}^2}{\mathrm{m}}$ und $A = 0{,}29\,\mathrm{mm}^2$ ergibt sich bei konstanten Strom von $I = 230\,\mathrm{mA}$

(vgl. Abbildung 4.1) nach Gleichung (4.6) ein Spannungsabfall für eine stromgeführte Datenleitung mit einer Länge von 200 m und einer doppelt rückgeführter Masseleitung von

$$\Delta U = U_{\text{TR\_4,stat}} - U_{\text{TR\_3,stat}}$$

$$= U_{6,\text{stat}} - U_{5,\text{stat}}$$

$$= U_{4,\text{stat}} - U_{5,\text{stat}} \quad \text{mit} = U_{6,\text{stat}} - U_{4,\text{stat}} = 0\ V$$

$$= \Delta U_{\text{Datenleitung}} + \Delta U_{\text{Masseleitung}} = 2{,}4\ \text{V} + 1{,}2\ \text{V} = 3{,}6\ \text{V}.$$

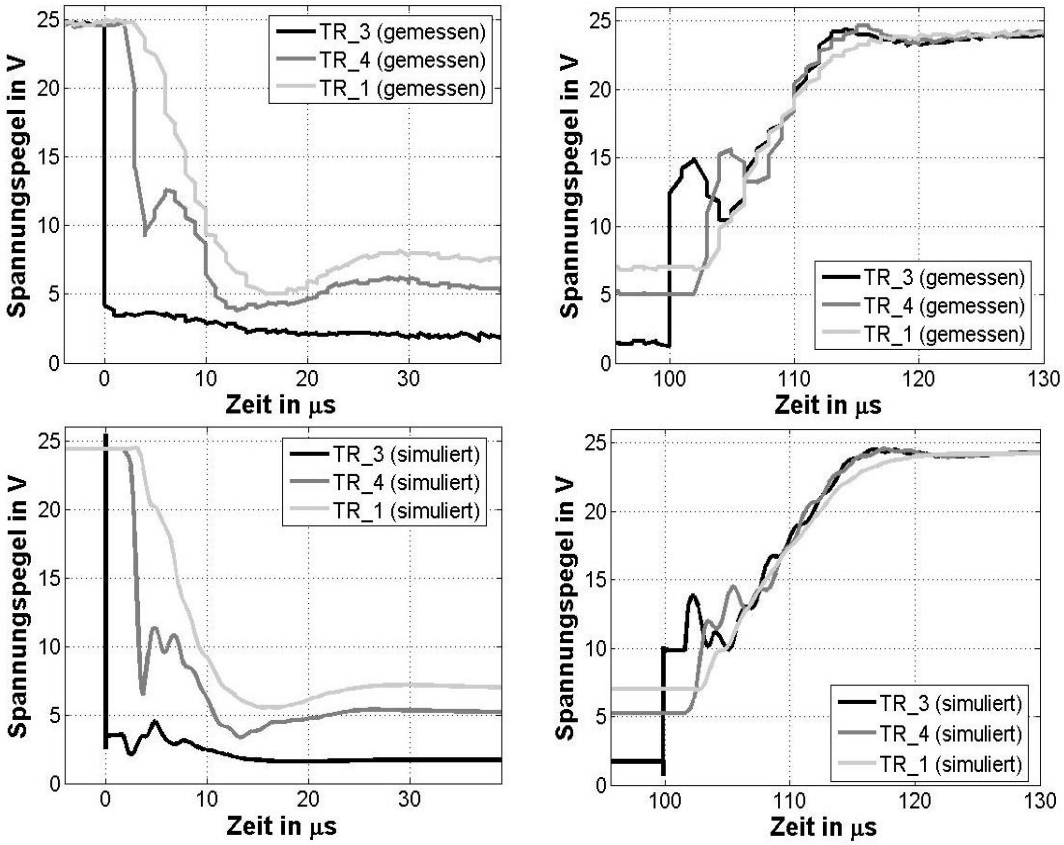


Abbildung 4.9: Fallende Flanken (links) und steigende Flanken (rechts) im zeitlichen Signalverlauf, gemessen (oben) und simuliert (unten) bei aktiven Transceiver TR_3

Signallaufzeit

Die Verzögerungen durch die Signallaufzeit sind im Verlauf der steigenden und fallenden Flanke erkennbar und liegen im Bereich des gemessenen bzw. rechnerisch bestimmbaren Wertes. In der Annahme eines Verkürzungsfaktors des Busmediums von $\frac{v}{c} = 0{,}66$ ergibt sich nach Gleichung (4.7) eine Signallaufzeit von $T_D = 2$ µs für eine Leitungslänge von 400 m zwischen TR_3 und TR_4.

$$T_{\mathrm{D}} = \frac{l}{0{,}66\ \mathrm{c}} \tag{4.7}$$

Entladen der Leitungskapazität

Mit Hilfe der Lösung der allgemeinen Differentialgleichung für die Bestimmung der Ladung an einer elektrischen Kapazität

$$\frac{\mathrm{d}Q}{\mathrm{d}t} = I(t) = C \cdot \frac{\mathrm{d}U}{\mathrm{d}t} \tag{4.8}$$

lässt sich die Zeit bis zur nahezu vollständigen Entladung (0,7 % von $U_{\max}$) des Kondensators über die Zeitkonstante $\tau = RC$ zu $T_{\mathrm{S}} = 5\tau$ berechnen. Für den Spannungsverlauf an TR_4 während des Zustandswechsels in den dominanten Zustand lässt sich die Entladezeit vereinfacht mittels der Entladung eines Kondensators über den aktiven Transceiver TR_3 bestimmen: $T_{\mathrm{S}} = 5\tau = 5 \cdot (R' \cdot\ 400\ \mathrm{m} \cdot C' \cdot 200\ \mathrm{m}) = 1{,}3\ \mu\mathrm{s}$ bei einem Kapazitätsbelag von $C' = 55\ \frac{\mathrm{pF}}{\mathrm{m}}$ und einem Widerstandsbelag von $R' = 0{,}05\ \frac{\Omega}{\mathrm{m}}$. Die gemessenen und simulierten Ergebnisse stimmen näherungsweise mit dem analytisch ermittelten Wert überein.

Die untersuchten Kennwerte für das Verhalten von Spannungsabfall, Signallaufzeit und Flankensteilheit sind sowohl für die Simulation als auch für die Messung nahezu identisch und entsprechen in etwa den analytischen Berechnungen. Die Spannungsschwankungen an der fallenden und steigenden Flanke zeigen sowohl für die Simulation als auch für die messtechnische Analyse qualitativ gleiche Verläufe. Die Ursache dafür ist das Reflexionsverhalten an den aktiven sowie an den sich im Leerlauf befindlichen Buskopplern, insbesondere bei langen Stichleitungen.

4.1.5 Einfluss der Teilnehmerzahl

Da in einem SmallCAN-System viele Busteilnehmer in einem Segment betrieben werden können, wird aufgrund unvermeidbarer Fehlanpassungen an passiven und aktiven Verbindungsstellen und aufgrund der erhöhten Anzahl parasitär wirkender Bauteilkomponenten die Signalqualität von vielen Faktoren beeinflusst. Durch Erprobungen ist sicherzustellen, dass die Signalqualität auch für maximal ausgeprägte Systemstrukturen hinreichend gut ist. Die Signalqualität zeichnet sich u. a. durch einen möglichst hohen Störabstand SNR bzw. einen möglichst minimalen Spannungsabfall im stationären Bereich sowie durch eine möglichst minimale Einschwingzeit nach einem Flankenwechsel aus.

In diesem Abschnitt werden die Simulationsergebnisse für verschiedene komplexe Ausprägungen des Bitübertragungssystems vorgestellt und der Einfluss auf die Signalqualität für den Fall einer steigenden Teilnehmeranzahl erläutert.

In Abbildung 4.10 werden die Signalverläufe für eine minimal sowie maximal ausgeprägte Topologie gemäß Abbildung 4.2 und Abbildung 4.3 für den Fall eines direkten Ansprechens des Ausgangstransistors dargestellt. Die Spannung zwischen der Datenleitung und GND wurden an drei Beobachtungspunkten im System ermittelt: in der Nähe des aktiven Buskopplers TR_3 bzw. TR_B991 (schwarz), in der Nähe der Energieversorgungseinheit bei Buskoppler TR_1 (dunkelgrau) und am Ende des Netzwerkes am Buskoppler TR_2 bzw. TR_495 (hellgrau). Für das minimale System, bestehend aus drei Buskopplern bei einer Gesamtkabellänge von 3 m, zeigt sich ein idealer und übereinstimmender Verlauf der Signale mit steilem Flankenverlauf und geringem Spannungsabfall über den Leitungen. Aus den Signalverläufen für die maximale Topologie (1000 Buskoppler, 1000 m Leitungslänge) ist zu beobachten, dass bedingt durch die erhöhte Leitungskapazität die Signalflanken deutlich abgeschwächt sind. Weiter sind der durch den Leitungswiderstand hervorgerufene Spannungsabfall zwischen den Datenleitungen und ein Rückschwingverhalten in der abfallenden Flanke zu erkennen, jedoch erreichte in allen Fällen (Variation der physikalischen Topologie) das Spannungssignal nach der Leitungslaufzeit und dem sich anschließenden Einschwingvorgang frühzeitig die Schwellenspannung von 13 V. Es hat sich gezeigt, dass im Wesentlichen die steigende Leitungslänge ein dominierender Einflussfaktor auf das Signalverhalten ist. Die steigende Anzahl der an dem Leitungsstrang angeschlossenen Teilnehmer wies keinen erkennbaren Einfluss auf die Eigenschaft der Signalqualität auf.

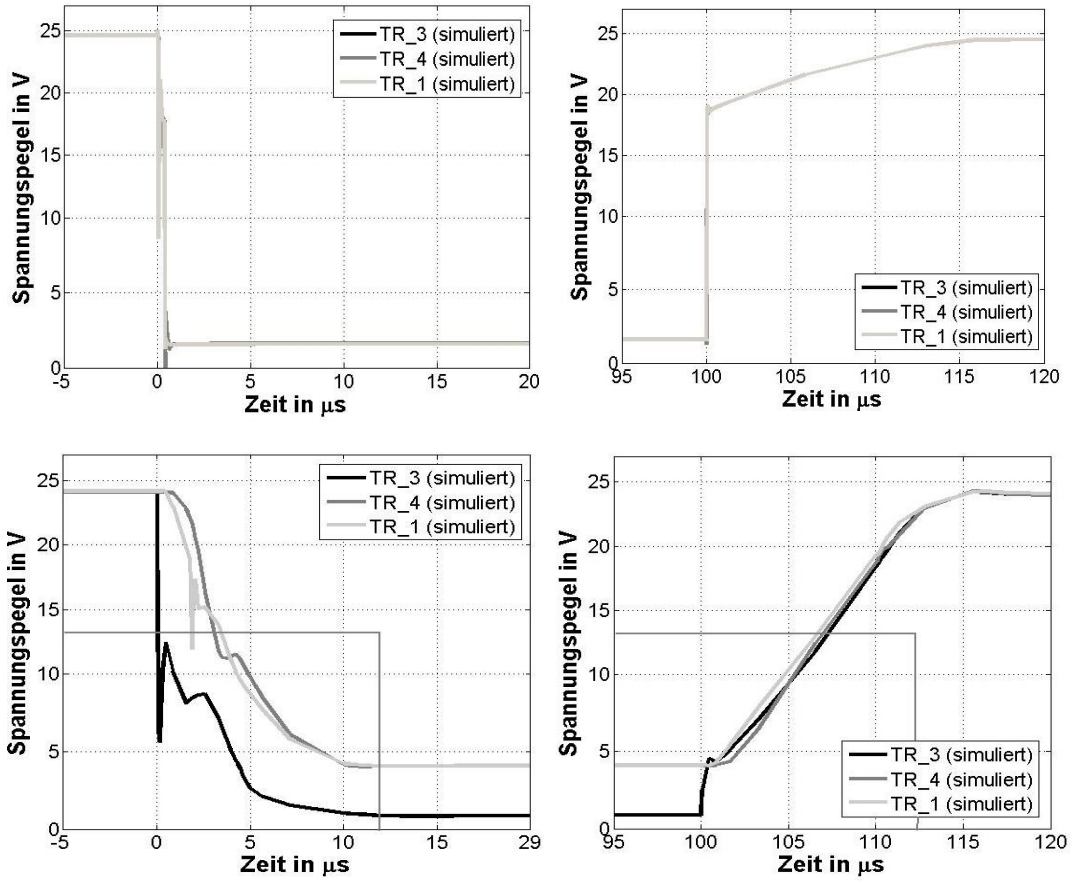


Abbildung 4.10: Simulationsergebnisse für eine minimale (oben) und eine maximale (unten) Topologie bei ungesteuertem Schalten des Ausgangstransistors

In Abbildung 4.11 werden für selbige Topologien aus Abschnitt 4.1.5 die Simulationsergebnisse unter Berücksichtigung einer kontrolliert verzögerten Ansteuerung des Ausgangstransistors, d. h. bei einem veränderten Modell der Transceiver-Schaltung BUS_TRANSCEIVER, dargestellt. Dabei führt ein verlangsamtes Durchschalten des Ausgangstransistors zu einem verlangsamten Flankenwechsel des Bussignals, wodurch zum Einen das Reflexionsverhalten und zum Anderen die Störaussendung (siehe Abschnitt 4.3.2) deutlich minimiert werden. Sowohl für die minimal als auch für die maximal ausgeprägte Kommunikationsarchitektur weisen die Verläufe der Spannungen ein ähnliches Verhalten auf. Beide Schaltungsvarianten des Transceivers erfüllen auch bei maximal ausgeprägter Topologie die Anforderungen an eine ausreichende Signalqualität hinsichtlich des Einschwingverhaltens und des Spannungsabfalls (vgl. Abbildung 4.1).

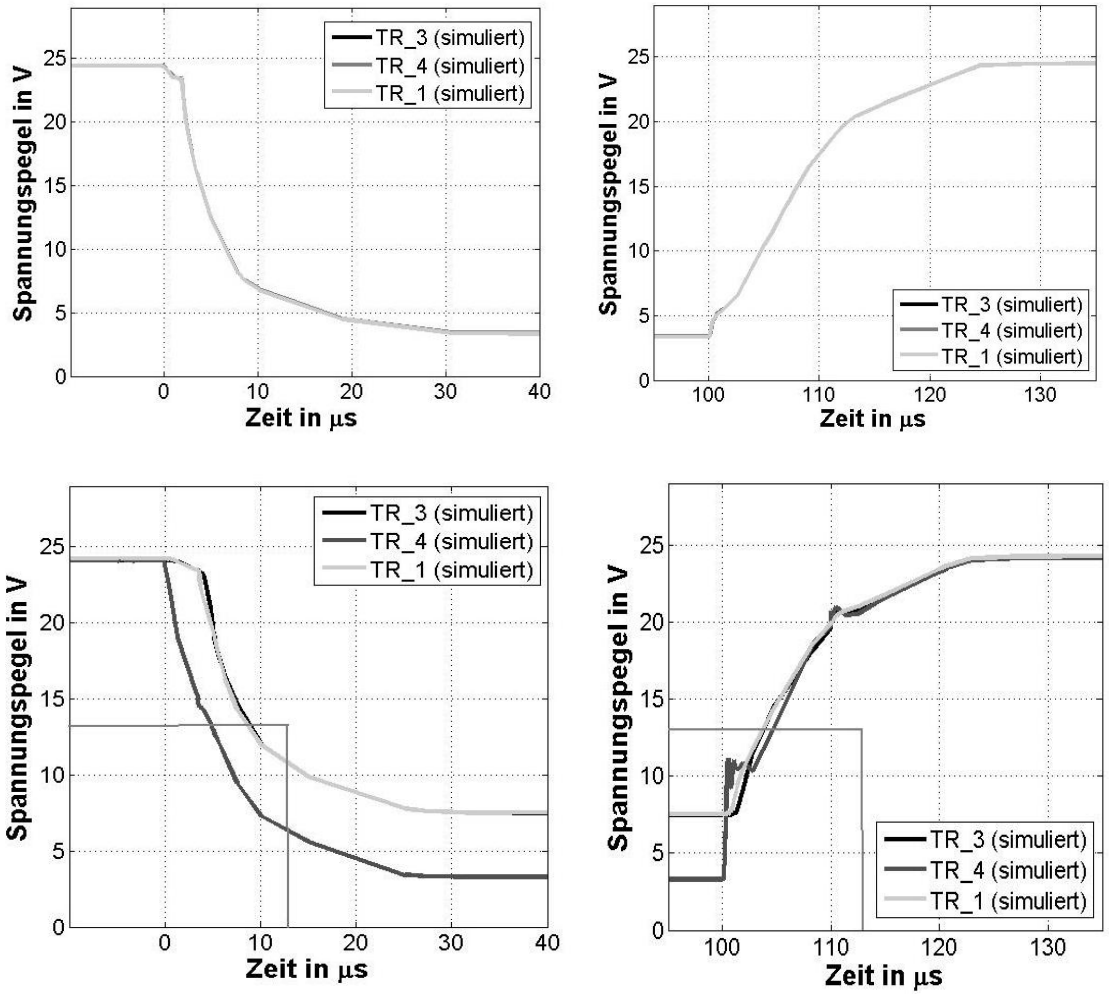


Abbildung 4.11: Simulationsergebnisse für eine minimale (oben) und eine maximale (unten) Topologie bei gesteuertem Schalten des Ausgangstransistors

4.1.6 Einfluss räumlicher Ausdehnungen

Bei steigender Komplexität des Bitübertragungssystems zeigt sich im direkten Vergleich der Signalverläufe an zwei unterschiedlichen Beobachtungspunkten auf der Datenleitung, dass die Einschwingzeit weit entfernt vom Sendeknoten größer ist als in der Nähe des sendenden Knotens. In Abbildung 4.12 wird dieses Verhalten unter der Berücksichtigung dargestellt, dass der sendende Knoten nah an der Versorgungseinheit angeordnet ist wodurch sich insgesamt ein minimaler Spannungsabfall auf den Datenleitungen einstellt. Die Einschwingzeit am Ende des Netzwerks ist für die maximale Ausdehnung etwa 10-mal größer als nah am Sender und führt zu einem verspäteten Unterschreiten der sicheren Schwellenspannung von 13 V nach ca. 2,2 µs. Dennoch wird die Schwellspannung innerhalb der maximalen Einschwingzeit von 11,9 µs erreicht. Während sich nah am sendenden Buskoppler der stationäre Zustand

innerhalb von 2 bis 3 µs einstellt, wird dieser am Ende des Netzwerkes erst nach ca. 16 µs erreicht.

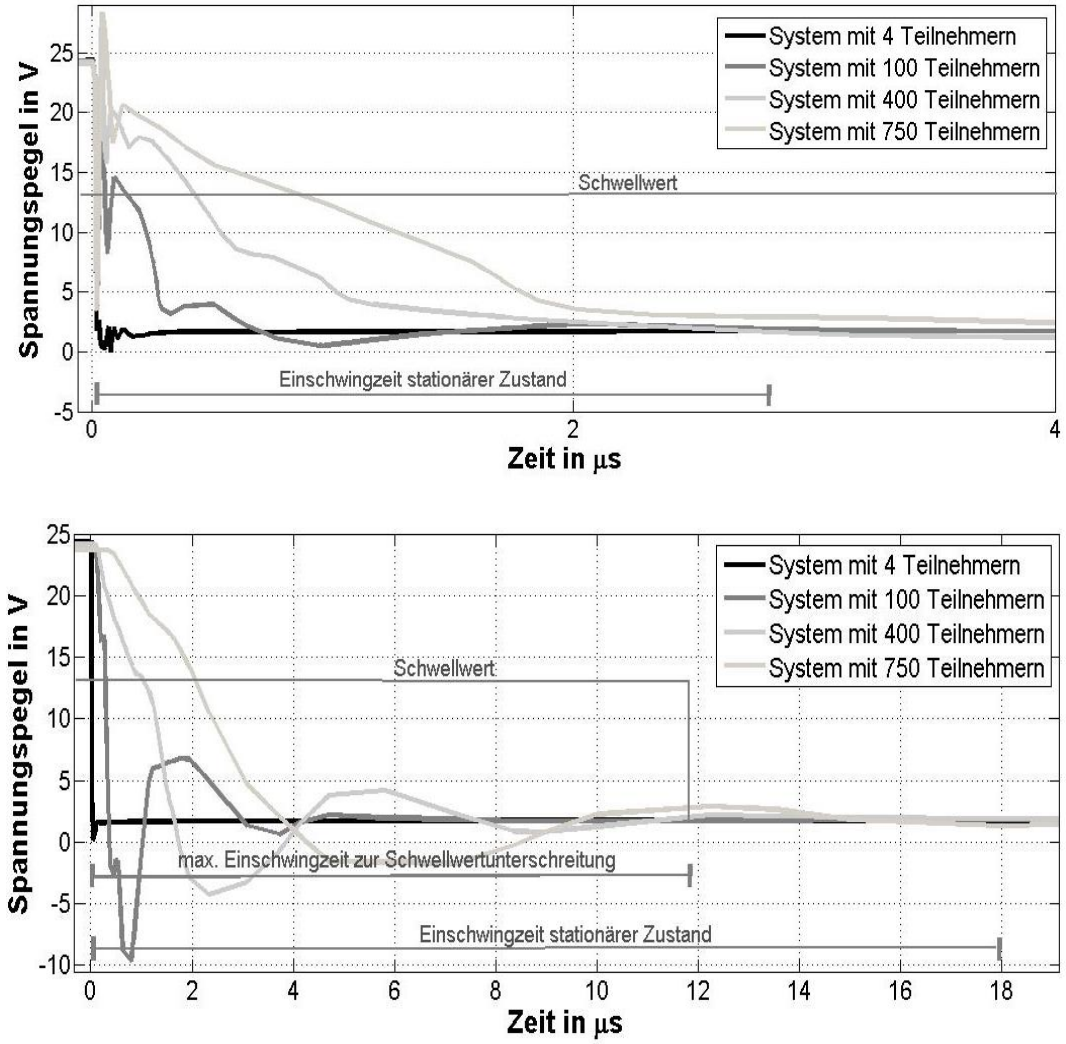


Abbildung 4.12: Fallende Flanke im zeitlichen Signalverlauf simuliert nah an der Versorgungseinheit und am sendenden Buskoppler (oben) sowie am Ende des Netzwerkes (unten) für unterschiedlich ausgedehnte Bustopologien

4.1.7 Einfluss von Stichleitungen

Um den Installationsaufwand möglichst gering zu halten und eine flexible Skalierbarkeit und Änderbarkeit des physikalischen Aufbaus zu gewährleisten, sind Terminierungskonzepte wie die Verwendung von Abschlusswiderständen für das Bitübertragungssystem von SmallCAN-System nicht vorgesehen. Der Verzicht auf solche reflexionsvermeidendenden Maßnahmen erfordert jedoch eine detaillierte Untersuchung der Auswirkungen auf die Signalqualität, insbesondere wenn das Übertragungsnetz längere Stichleitungen aufweist. Aus den in Abbildung 4.13

dargestellten Simulationsergebnissen wird ersichtlich, dass eine Erhöhung der Stichleitungslänge zu stärkerem Reflexionenverhalten führt. Im Fall langer Stichleitungen weist der Signalverlauf während der fallenden Flanke zunächst noch ein frühzeitiges Unterschreiten der Schwellenspannung von 13 V auf. Nach etwa 1 µs kommt es jedoch zu einem signifikanten Rückschwinger, der den Signalpegel in den Bereich des Schwellenwertes ansteigen lässt. Die Spannung klingt aber ausreichend schnell wieder ab, so dass sich zum ersten Abtastzeitpunkt, beispielsweise bei $\frac{1}{4}$ der Bitzeit, ein Spannungswert unterhalb der tolerierbaren Schwellspannung einstellt.

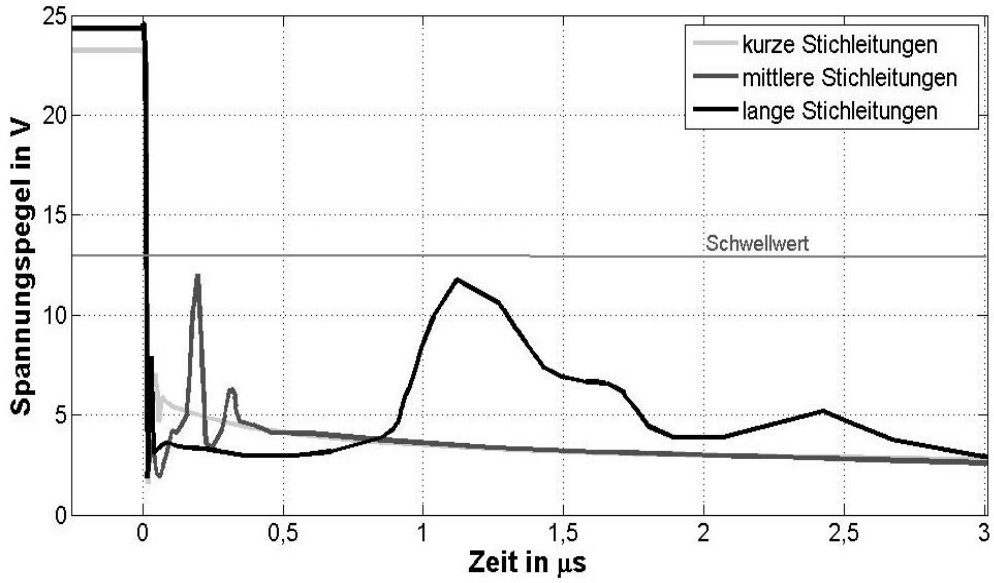


Abbildung 4.13: Rückschwingverhalten bei unterschiedlich ausgeprägten Stichleitungslängen, simuliert in der Nähe des sendenden Buskopplers

4.2 Störfestigkeit

Neben der Erfüllung der Signalqualität zur Gewährleistung einer sicheren Kommunikation der Geräte untereinander müssen diese vor erhöhten Störimpulsen geschützt werden, die beispielsweise durch eingekoppelte Überspannungen bei Blitzentladungen oder durch Einwirkung von Schaltvorgängen benachbarter Geräteeinrichtungen entstehen. Im Allgemeinen ist die Einhaltung der Störfestigkeit für Versorgungseingänge einfach zu realisieren. Der Schutz von Datenleitung kann aufgrund der dynamischen Signalführung jedoch einige Herausforderungen mit sich bringen: insbesondere für ausgeprägte Systeme mit bis zu 1000 Teilnehmern kann aufgrund des kapazitiven Anteils der Schutzelemente, platziert am Signaleingang eines jeden Teilnehmers, das Verhalten des Bitübertragungssystems signifikant beeinflusst werden (reduzierte Flankensteilheit) bzw. der benötigte Strom zum Umladen steigt mit zunehmender Gesamtkapazität an. Die zu berücksichtigende Gesamtkapazität ergibt

sich demnach aus der Summe der parasitären Kapazitäten sämtlicher Schutzelemente addiert zu der Kapazität der Busleitung, die bei großen Leitungslängen bereits signifikant hoch sein kann. Zwar stehen Schutzelemente zur Verfügung, die aufgrund ihrer geringen kapazitiven Wirkung für Datenübertragungssysteme als geeignet erscheinen, jedoch weisen viele standardisierte Bauelemente kleinere Begrenzungsspannungen auf als für das System SmallCAN gefordert (vgl. Betriebsspannungen aus Tabelle 9). In Tabelle 10 sind die für das System SmallCAN als geeignet bewerteten Maßnahmen für den Schutz der Datenleitung gegenübergestellt und relevante Merkmale sowie die Ergebnisse einer normativen Störfestigkeitsprüfung aufgezeigt. Durch eine Reihenschaltung zweier Dioden-Arrays konnte bei geringer kapazitiver Beeinflussung ein ausreichend hoher Prüfschärfegrad für den Gebrauch des Systems in Wohnbereichen, Geschäfts- und Gewerbebereichen sowie Kleinbetrieben erzielt werden. Ergänzend sei angemerkt, dass im Fall ausgeprägter Topologien durch die physikalische Verknüpfung sämtlicher Schutzelemente eine Mehrfachwirkung erzielt werden kann, wodurch eine Unterdrückung von Störimpulsen weiter begünstigt wird.

Tabelle 10: Maßnahmen zur Erhöhung der Störfestigkeit des SmallCAN-Buskopplers

	Maßnahmen			
	Einfache Supressor-diode	**Varistor**	**TVS-Diode**	**Dioden-Array**
	(WE-VE, 26VDC)	(WE-VS, 38VDC)	(LCDA24C)	(2x TClamp 1272S)
Merkmale				
Kapazität	++	--	+++	+++
Begrenzungs-spannung	+++	+++	++	++
Preis	+++	+++	++	+
Abmaße	+++	++	+	+
Erreichter Prüfschärfegrad	---	+++	-	++

von --- (ungeeignet) bis +++ (sehr geeignet)

4.3 Störaussendung

Zur Ermittlung der Ursachen einer erhöhten Störaussendung des Systems und zur Bewertung geeigneter Gegenmaßnahmen sind messtechnische und simulative Untersuchungen durchgeführt worden, die in den folgenden Abschnitten beschrieben und erläutert werden.

4.3.1 Einfluss von Filter und Schirmmaßnahmen

Zur Bewertung verschiedener Filter und Schirmmaßnahmen wurden Untersuchungen zur gestrahlten Störaussendung in einer TEM-Zelle (Transverse Electromagnetic Cell) im Frequenzbereich von 1 MHz bis 400 MHz durchgeführt. Die Wirksamkeit der TEM-Zelle, die als Messkammer eine definierte Messumgebung für kleinere Prüfvolumen bereitstellt, wurde mit Hilfe einer Nullmessung verifiziert. Als Filtermaßnahme wurde eine CAN-Drossel zur Unterdrückung der Störgrößenausbreitung eingesetzt. Wie aus Abbildung 4.14 zu ersehen ist, zeigt sich durch das Filterelement lediglich eine minimale Reduzierung der Störaussendung. Abgesehen davon ist der Einsatz von CAN-Drosseln als Filtermaßnahme im Vergleich zu deren Dämpfungswirkung insbesondere unter Berücksichtigung der Kosten für Systemausführungen mit vielen Busteilnehmern als nicht sinnvoll zu erachten.

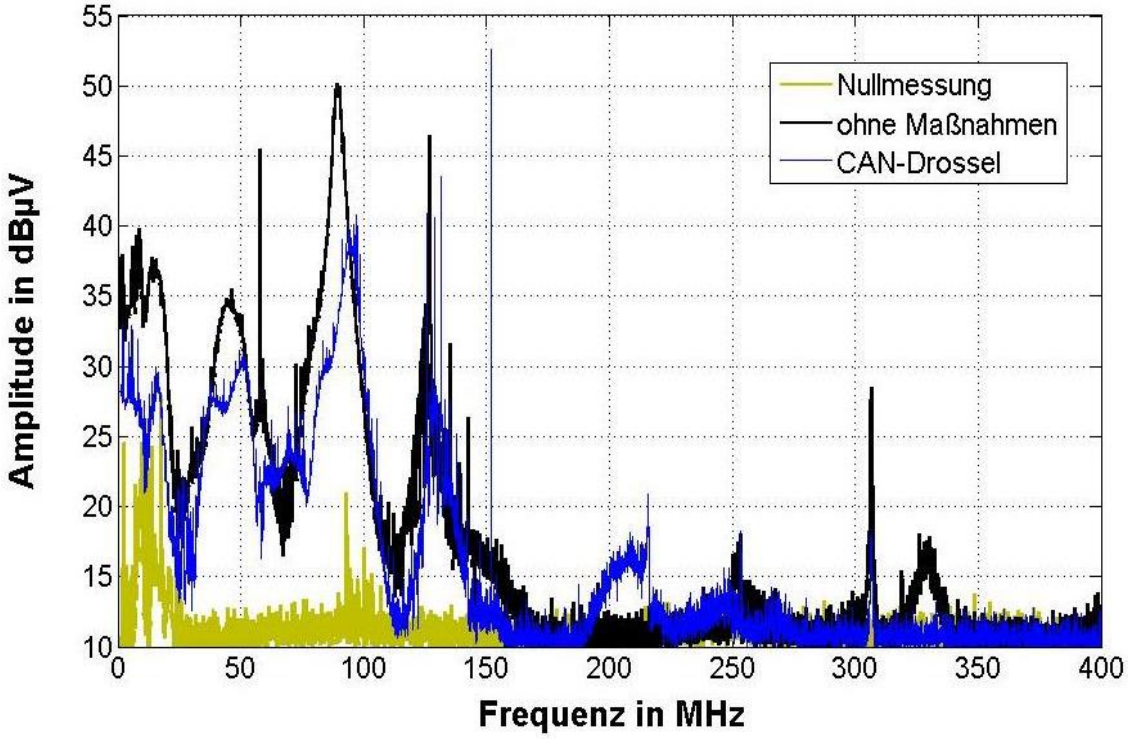


Abbildung 4.14: Einfluss von CAN-Drosseln als Filtermaßnahme auf die Störaussendung, gemessen in einer TEM-Zelle

Es wurden weitere Messungen zur Bewertung möglicher Schirmmaßnahmen durchgeführt. Aus Abbildung 4.15 ist zu entnehmen, dass die Störaussendung im Wesentlichen leitungsgebundene Ursachen haben, da eine Abschirmung der Platinenkomponenten nur wenig Wirkung zeigte. Mit Auflegung des Kabelschirms auf Erdpotenzial und unter Verwendung eines paarweise stark verdrillten Kabels (Twisted Pair) als Busmedium konnte die Störaussendung zwar verbessert werden, für den praktischen Einsatz sind diese Maßnahmen jedoch weniger geeignet, da der Verlegeaufwand für ein stark verdrilltes Kabel mit Schirm und die Kosten für die Bereitstellung von Schirmklemmen deutlich ansteigen würden. Weiterhin wäre die Flexibilität in der Wahl des Installationskabels eingeschränkt.

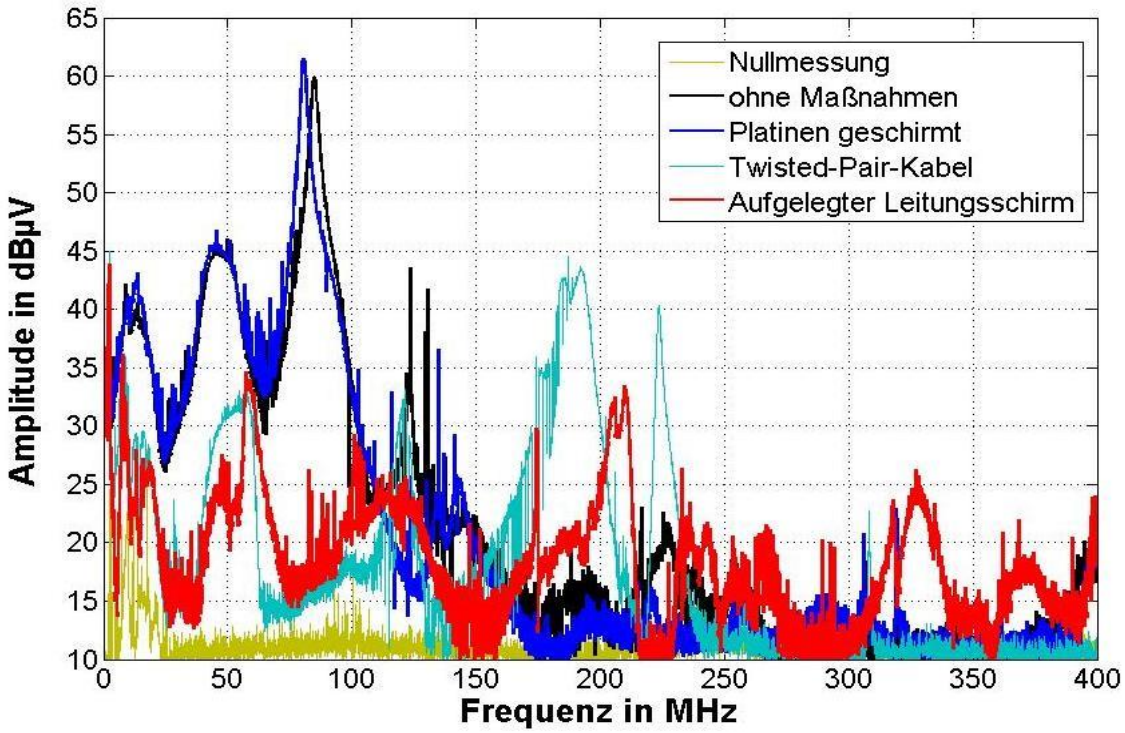


Abbildung 4.15: Einfluss verschiedener Schirmmaßnahmen auf die Störaussendung, gemessen in einer TEM-Zelle

4.3.2 Einfluss der Signalflankensteilheit

Eine deutliche Verbesserung der Störaussendung konnte durch eine Reduzierung der Flankensteilheit beim Zustandswechsel auf der Signalleitung erreicht werden. Diese Maßnahme kann durch ein gezielt verzögertes Durchschalten des Schalttransistors (kontrollierte Annäherung an Schwellwert) erreicht werden. Ein verzögertes Schaltverhalten bedingt unmittelbar eine Reduzierung der Signalflanken auf dem Bus und begünstigt dadurch die Störaussendung. In Abbildung 4.16 und Abbildung 4.17 werden die Prüfergebnisse aus der Modenverwirbelungskammer sowie aus der

Schirmkabine mit variierender Positionierung des Prüfaufbaus dargestellt. Die Messresultate zeigen deutlich die Wirksamkeit dieser Maßnahme auf.

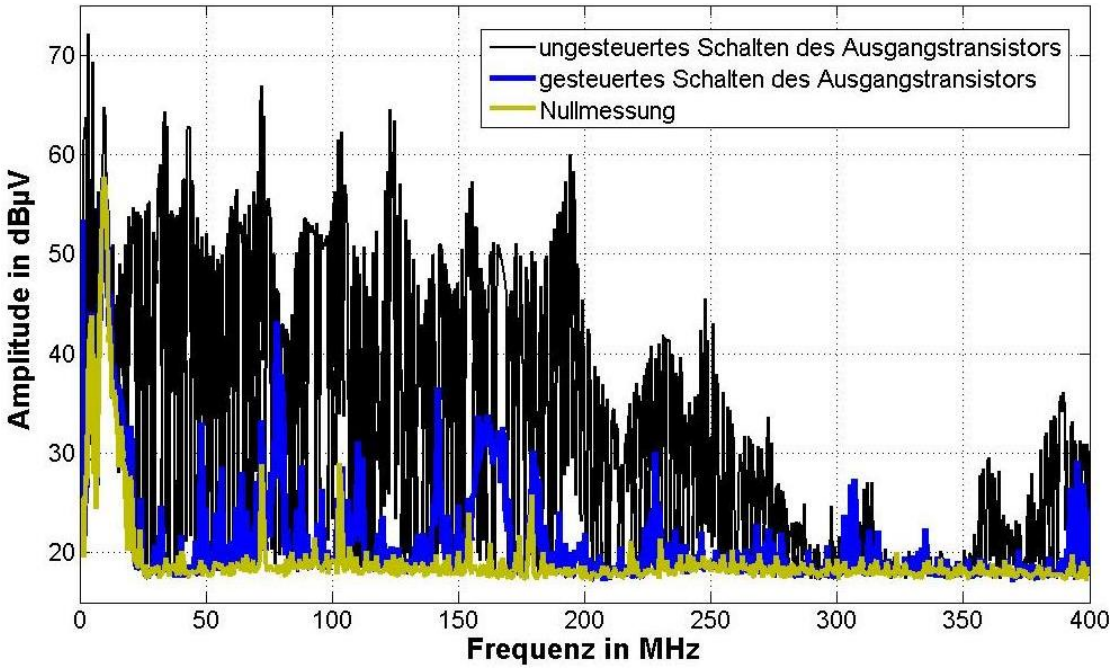


Abbildung 4.16: Störaussendungsmessung in Modenverwirbelungskammer für verschiedene Ansteuerungsarten des Ausgangstransistors

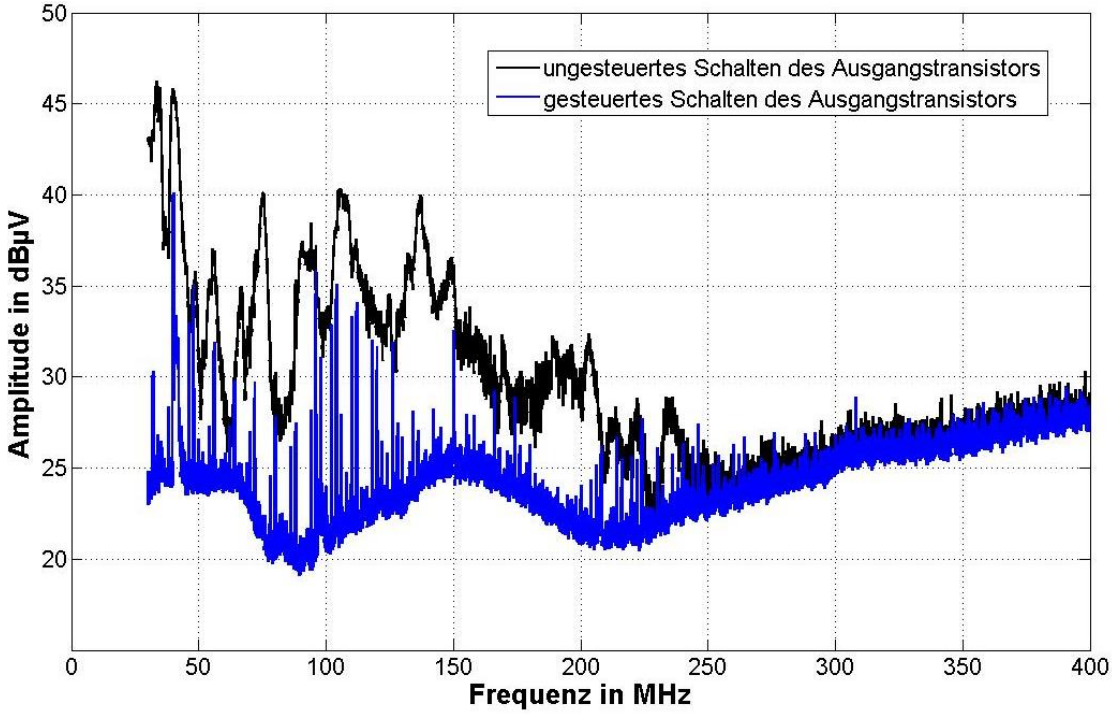


Abbildung 4.17: Normative Messung zur Störaussendung in Schirmkabine für verschiedene Ansteuerungsarten des Ausgangstransistors

Ergänzend zeigt Abbildung 4.18 die simulierten Frequenzgänge der Bussignale aus Abbildung 4.10 und Abbildung 4.11 für das gesteuerte (verzögerte) und ungesteuerte (direkte) Schalten des Ausgangstransistors des aktiven Transceivers. Die veränderten Ausprägungen der Signalamplituden, insbesondere bei höhere Frequenzen, verdeutlichen die Unterschiede in der Flankensteilheit der Signale.

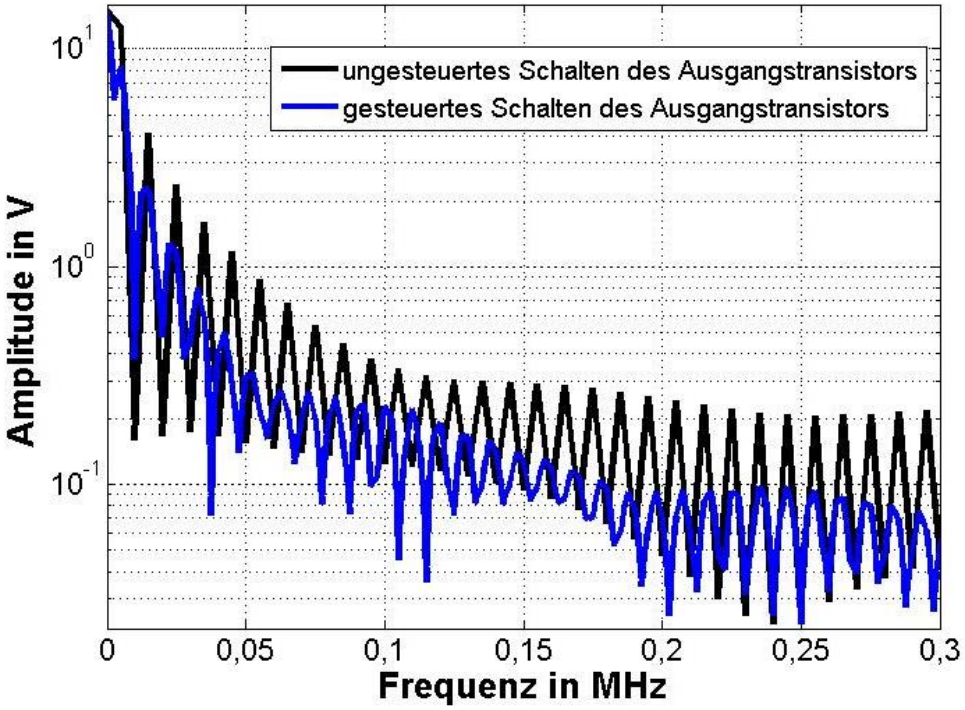


Abbildung 4.18: Frequenzgang-Darstellung simulierter Signalverläufe für die verschiedenen Ansteuerungen des Ausgangstransistors

Die untersuchten Maßnahmen zur Störaussendung und ihr Einfluss auf die Qualitätsmerkmale des Gesamtsystems sind in Tabelle 11 zusammengefasst. Durch das kontrollierte Ansteuern des Ausgangstransistors konnte eine deutliche Verbesserung der Störaussendung erreicht werden ohne den Verlegeaufwand und die Materialkosten zu erhöhen. Bedingt durch die mit dieser Maßnahme verbundene reduzierte Flankensteilheit ist lediglich die Leistungsfähigkeit des Übertragungssystems in Form einer begrenzteren Datenrate eingeschränkt. Bei maximal ausgeprägten Architekturen gilt diese Einschränkung jedoch generell (vgl. Abschnitt 4.1.5). Die weiteren gelisteten Maßnahmen haben einem Anstieg der Materialkosten sowie eine Erhöhung des Verlegeaufwandes zur Folge und führen im Vergleich zur Reduktion der Flankensteilheit nur zu einer geringen Verbesserung der Störaussendung.

Tabelle 11: Bewertete Maßnahmen zur Reduzierung der Störaussendung für SmallCAN

	Maßnahmen				
	Filter	**Schirmung**			**Vermeidung**
Merkmals-beeinflussung	CAN-Drossel	Geschirmte Platine	Aufgelegter Leitungs-schirm	Geschirmte Adernpaare	Gesteuerter Ausgangs-transistor
Störaussendung	+	+	++	++	**+++**
Verlegeaufwand	+/-	--	---	-	**+++**
Materialkosten	---	---	-	--	**++**
Abmaße	--	--	-	-	**+/-**
Bitstrom	++	++	++	++	**+/-**

von --- (stark negativ beeinflusst) bis +++ (stark positiv beeinflusst)

4.4 Rückführung der Analyseergebnisse in das Systemmodell

In diesem Kapitel wurde mit Hilfe einer SPICE-Simulation die Signalqualität des verteilten Automatisierungssystems SmallCAN untersucht, welches sich im Vergleich zu standardisierten Systemen wie CAN oder RS485 durch eine hohe strukturelle Skalierbarkeit eines Segmentes auszeichnet (Anforderung: ausreichende Signalqualität in einem Segment mit 1000 Teilnehmer bei 1000 m Leitungslänge). Für diese Fragestellung sind in Abbildung 4.19 die ermittelten Simulationsergebnisse zusammenfassend dargestellt. Für verschiedene maximal ausgeprägte physikalische Topologien blieben die Anforderungen an eine ausreichende Signalqualität erfüllt. Sowohl für eine erhöhte Anzahl an Teilnehmern als auch für räumlich ausgedehnte Systemstrukturen wurde in der Simulation weder der maximal tolerierbare Spannungsabfall von 7 V pro Ader noch die maximale erlaubte Einschwingzeit von 11,9 µs bis zur Schwellspannung von 13 V überschritten. Für alle geprüften Topologien wies der stationäre Leitungszustand im rezessiven Fall einen Wert größer 17,95 V und im Fall eines dominanten Pegels einen Wert kleiner 13 V auf. Damit konnte in der Simulation nachgewiesen werden, dass ein SmallCAN Segment mit 1000 vernetzen Teilnehmern bei einer Leitungslänge von 1000 m den Anforderungen an eine ausreichende Signalqualität genügen.

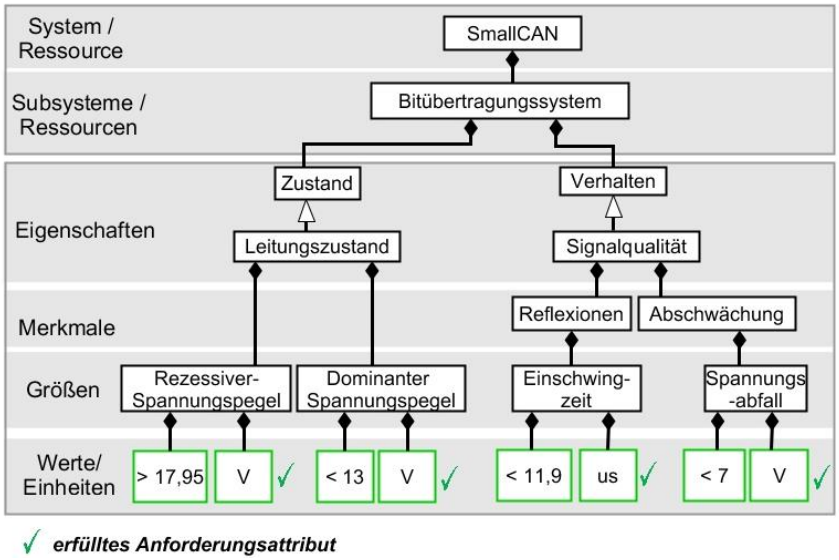


Abbildung 4.19: Ergebnisrückführung in das Systemmodell nach der Signalqualitätsanalyse

Weiterhin wurden in diesem Kapitel durchgeführte Analysen zur elektromagnetischen Verträglichkeit vorgestellt und verschiedene Maßnahmen zur Erhöhung der Störfestigkeit und zur Reduzierung der Störaussendung diskutiert. Auf Basis der Untersuchungen konnten geeignete Maßnahmen gefunden werden, durch die die normativen Anforderungen für den Gebrauch des Systems SmallCAN in Wohnbereich, Geschäfts- und Gewerbebereichen sowie Kleinbetrieben erfüllt sind. In Abbildung 4.20 sind die erreichten Prüfschärfegrade für das System SmallCAN unter Berücksichtigung der Prüfnormen [DIN EN 55016-2-3], [DIN EN 61000-4-4], [DIN EN 61000-4-5], [DIN EN 61000-4-6], [DIN EN 61000-4-3] und [DIN EN 61000-4-2] im Form der Attributhierarchie zusammengetragen.

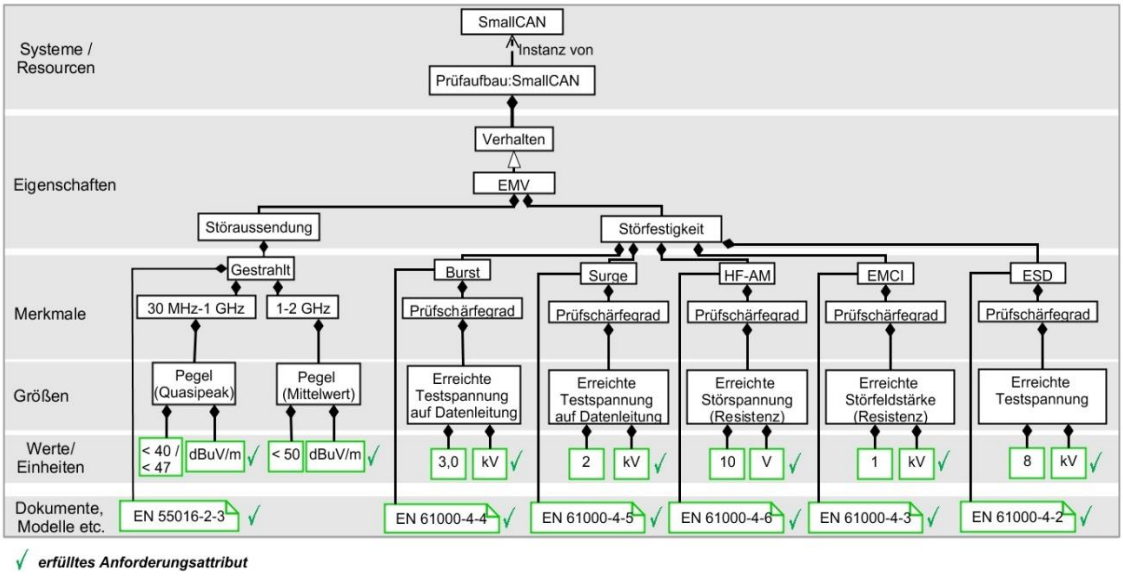


Abbildung 4.20: Ergebnisrückführung in das Systemmodell nach den Analysen zur elektromagnetischen Verträglichkeit

5 Modellierung und Analyse von Verzögerungs- und Laufzeiten

Das folgende Kapitel behandelt die Betrachtung des dynamischen Verhaltens verteilter Automatisierungssysteme. Das Verhalten ist im Wesentlichen durch die Wahl der Systemarchitektur, d. h. der Abbildung der Verarbeitungsfunktionen auf die Software und Hardware sowie deren Verknüpfungen untereinander, beeinflusst. Daraus ergeben sich folgende Problemstellungen:

- Mit der Aufteilung und Abbildung der zu erfüllenden Systemfunktionen auf das verteilte Hardwaresystem kommt es bei der Ausführung physikalisch bedingter Kommunikationsprozesse zu Konfliktsituationen und damit zu einem nichtdeterministischen Verzögerungsverhalten aufgrund der gleichzeitigen Anforderung der Ressource des Übertragungskanals.

- Um mit steigendem Funktionsumfang die Komplexität innerhalb des Anwendungssystems beherrschbar zu halten, werden zunehmend wiederverwendbare und modulare Programmkomponenten eingesetzt. Im Falle ihrer zyklischen Bearbeitung (vgl. Abbildung 5.7, unten) treten im Systemverbund Zyklusübergänge auf. Es kommt demnach bei der Datenübergabe zwischen den Programmkomponenten ebenfalls zu Verzögerungen aufgrund von Synchronisierungsprozessen.

Daher werden in Abschnitt 5.1 das Buslastverhalten und in Abschnitt 5.2 das Zyklusübergangsverhalten näher behandelt. Die der jeweiligen Problemstellung zuzuordnenden (Teil-)Systeme werden zunächst je durch ein Systemmodell gemäß der in Kapitel 3 vorgeschlagenen Vorgehensweise beschrieben. Auf der Basis dieser Systembeschreibung lässt sich eine Modellierung ausführbarer Petrinetzmodelle durchführen, die mathematisch beschrieben und validiert wurden. Desweiteren werden Ergebnisse aus den Simulationen der Petrinetzmodelle vorgestellt und diskutiert.
Um eine bestimmte Abfolge von verteilten Funktionen unter Berücksichtigung sämtlicher Verzögerungen, d. h. der Busauslastung und der Zyklusübergänge, darstellen und deren Laufzeit ermitteln zu können wird in Abschnitt 5.3 ein Vorgehen zur automatisierten Erstellung eines Petrinetzmodells vorgestellt, welches einen vollständigen Informationsprozess abbildet. Für eine weiterführende Erläuterung der Analyse eines vollständigen Prozessablaufes erfolgt die Anwendung des Vorgehens am Beispiel eines Funktionsablaufes aus Abschnitt 1.1.1.
Mit Hilfe der Simulationen eines verteilten Systems zur Erfassung dynamischer Eigenschaften kann u. a. eine frühzeitige Evaluierung von Entwurfsentscheidungen,

beispielsweise dem Aufdeckung von Defiziten in der Systemarchitektur erreicht werden [Diekhake/Schnieder 2015]. Weiterhin können die Ergebnisse in späteren Lebenszyklusphasen des Systems Verwendung finden. Beispielsweise lassen sich aus ermittelten Kennwerten zur Laufzeit des kausalen Prozessablaufes realitätsnahe Aussagen zu tolerierbaren Laufzeitschranken ableiten, die im operativen Betrieb des Automatisierungssystems überwacht werden können (Abschnitt 7.2.2).

5.1 Buslastverhalten

Die gemeinsame Nutzung der Kommunikationsressource des Busmediums erfordert für die Telegrammübertragung das Erlangen eines Sendesrechtes auf dem Übertragungskanal, welches für das betrachtete System SmallCAN über das CSMA/CA-Verfahren koordiniert wird. Durch das nichtdeterministische Verfahren kann es bei einer erhöhten Kanalauslastung zu signifikanten Verzögerungen für das Absenden niederpriorer Telegramme kommen, da eine Mehrfachbelegung des Kanals nicht möglich ist (Bisektionsweite = 1). In den Abschnitten 5.1.1 und 5.1.2 werden das der Problemstellung zugrundeliegende Systemmodell sowie das sich daraus ergebene Simulationsmodell exemplarisch vorgestellt. Weiterhin wird in Abschnitt 5.1.3 eine Validierung des Simulationsmodells behandelt. Bezogen auf die Ermittlung von Kommunikationsverzögerungen werden abschließend ausgewählte Simulationsergebnisse dargestellt. Die Verläufe der Verzögerungszeiten eines niederprioren Telegrammes bei verschiedenen Busauslastungen wurden ermittelt und grafisch dargestellt (Abschnitt 5.1.4). Desweiteren ist der Einfluss der Sendeart auf die Verteilungsverläufe untersucht worden (Abschnitt 5.1.5). In Abschnitt 5.1.6 wird die Beeinflussung der Gerätezahl auf die Verteilung der Verzögerungszeiten diskutiert.

5.1.1 Systemmodell

Das Systemmodell aus Abbildung 5.1 fasst folgende für das Buslastverhalten als relevant erachtete Subsysteme zusammen:

- das funktional betrachtete *Kommunikationssystem*
- das funktional betrachtete *Anwendersystem* (vgl. Abbildung 3.11)
- das räumlich betrachtete *Datenübertragungssystem*
- das funktional und technologisch betrachtete *Gerätesystem.*

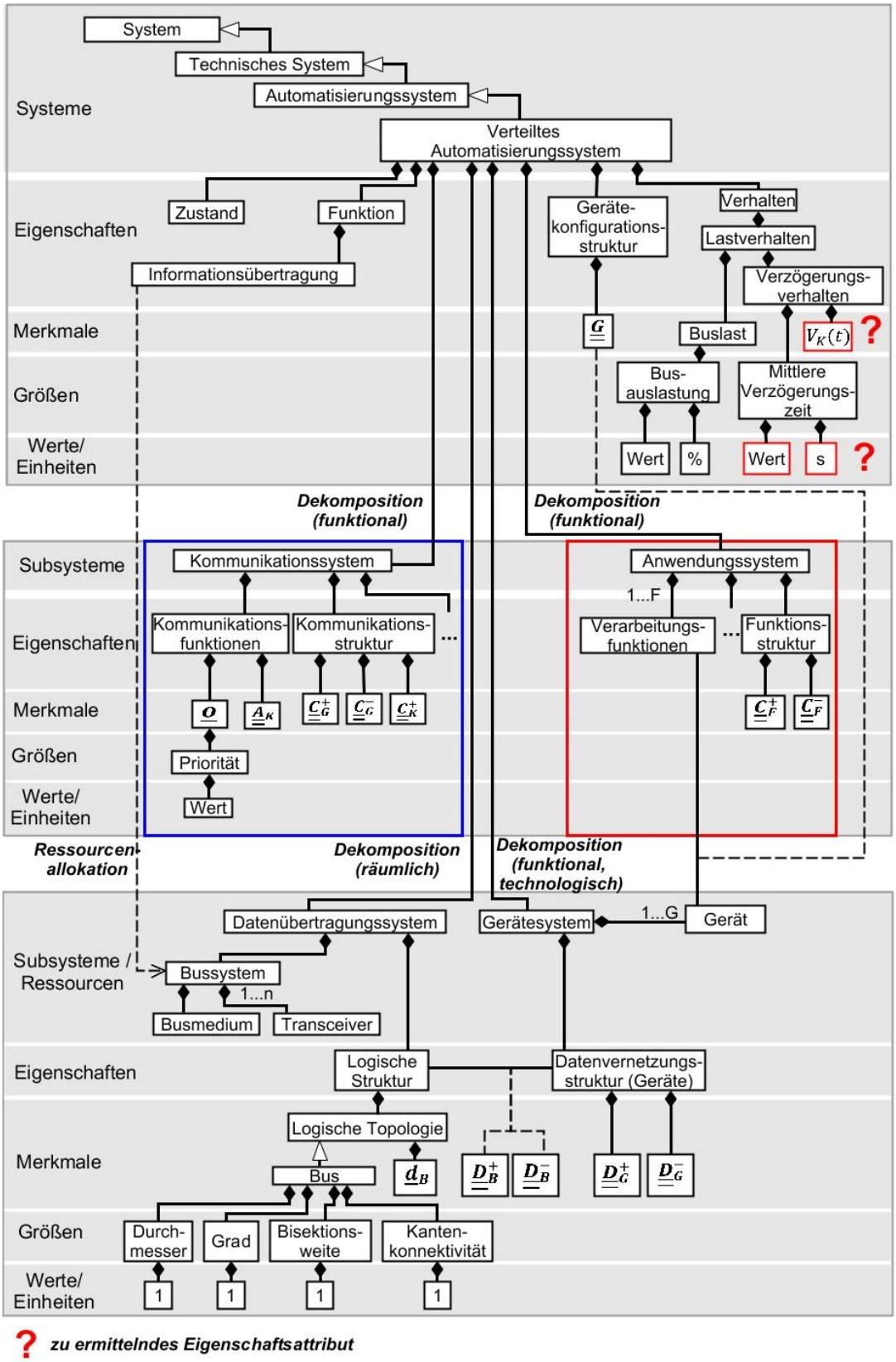


Abbildung 5.1: Ganzheitliches Systemmodell angepasst an die Problemstellung des Buslastverhaltens in einem verteilten Automatisierungssystem mit funktional betrachteten Kommunikationssystem (blau) und funktional betrachteten Anwendungssystem (rot) als Dekompositionen eines verteilten Automatisierungssystems.

Die Zuordnung von verarbeitenden Funktionen auf die dezentral angeordneten Geräte erfolgt über die *Gerätematrix* nach [Kiefer 1995] bzw. einer Allokationsmatrix [Schnieder 2010: 138].

$$\underline{\boldsymbol{G}} = (g_{ij}) \text{ mit } g_{ij} = \begin{cases} 1, & \text{wenn Funktion } i \text{ auf Gerät } j \text{ implementiert ist} \\ 0, & \text{sonst} \end{cases} \tag{5.1}$$

In UML stellt die Gerätematrix ein Attribut einer Assoziationsklasse dar, welche die Assoziation zwischen der Klasse der *Geräte* und der Klasse der *verarbeitenden Funktionen* beschreibt.
Durch die Verteilung der verarbeitenden Funktionen auf die dezentral angeordneten Geräte, sind *Kommunikationsfunktionen* erforderlich, die dem funktional betrachteten Kommunikationssystem zugewiesen werden. Für eine Kommunikationsfunktion als Ereignis des Ablegens eines Telegrammes auf dem Bus können folgende temporale Merkmale definiert werden.

1. Der *Prioritätenvektor*, der den Kommunikationsfunktionen einen Prioritätswert zuteilt:

   $$\underline{\boldsymbol{o}} = (o_i) \text{ mit } o_i = \text{Prioritätswert der Kommunikationsfunktion } i \tag{5.2}$$

2. Die *Verhaltensmatrix* zur Beschreibung der Versandart der Kommunikationsfunktionen über eine Verteilungsfunktionen *dist* und deren Parameter *par*:

   $$\underline{\boldsymbol{A}}_{\boldsymbol{K}} = (a_i) = \begin{matrix} \text{Verteilung} \\ \\ \end{matrix} \overset{\begin{matrix} \text{Parameter 1} \end{matrix}}{\begin{pmatrix} dist_1 & par1_1 & par2_1 \\ \vdots & \vdots & \vdots \\ dist_i & par1_i & par2_i \end{pmatrix}} \begin{matrix} \\ \text{Kommunikations-} \\ \text{funktionen} \end{matrix} . \tag{5.3}$$

Für das funktional betrachtete Anwendersystem lässt sich die *Funktionsstruktur*, d. h. die funktionale Darstellung der Datenflüsse zwischen den einzelnen verarbeitenden Funktionen, über die Inzidenzmatrizen $\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F}}^{+}$ und $\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F}}^{-}$ beschreiben, wobei die Spalten die Funktionen und die Zeilen die verknüpfenden Daten repräsentieren [Kiefer 1995].

Unter Berücksichtigung der logischen Strukturbeschreibung des Datenübertragungssystems (*Bisektionsweite* = 1), dem daraus abgeleiteten *Busübertragungsvektor* nach [Kiefer 1995]

$$\underline{\boldsymbol{d}}_{\boldsymbol{B}} = (d_i) \text{ mit } d_i = \underline{\underline{\boldsymbol{D}}}_{\boldsymbol{G},ij}^{-} \left(\underline{\mathbf{1}} - \underline{\boldsymbol{I}}_j\right) \wedge \underline{\underline{\boldsymbol{D}}}_{\boldsymbol{G},ij}^{+} = 1 \tag{5.4}$$

und der gerätebezogenen logischen *Datenvernetzung*

$$\underline{\underline{\boldsymbol{D}}}_{\boldsymbol{G}}^{-} = \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F}}^{-}\underline{\underline{\boldsymbol{G}}} \tag{5.5}$$

$$\underline{\underline{\boldsymbol{D}}}_{\boldsymbol{G}}^{+} = \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F}}^{+}\underline{\underline{\boldsymbol{G}}} \tag{5.6}$$

lassen sich die Relationen zwischen den Verarbeitungsfunktionen und den auf dem Bus zur Verfügung stehenden Daten (Telegrammen) über die folgenden Kommunikationsmatrizen für *Multicast*-Verbindungen zwischen den Teilnehmern beschreiben:

$$\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G}}^{+} = \underline{\underline{\boldsymbol{C}}}^{(\mathbf{1}:\boldsymbol{n})} = \left(c_{ij}\right) \text{ mit } c_{ij} = \begin{cases} 1, & \text{wenn } \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F},ij}^{+}=1 \ \wedge \underline{\boldsymbol{d}}_{\boldsymbol{B},i}\neq 0 \\ 0, & \text{sonst} \end{cases} \tag{5.7}$$

und

$$\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G}}^{-} = \left(c_{ij}\right) \text{ mit } c_{ij} = \begin{cases} 1, & \text{wenn } \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F},ij}^{-}=1 \ \wedge \underline{\boldsymbol{d}}_{\boldsymbol{B},i}\neq 0 \\ 0, & \text{sonst} \end{cases} . \tag{5.8}$$

Die Matrizen aus (5.5) und (5.6) repräsentieren die Zuteilung aller im System vorhandenen Daten zu den entsprechenden Geräten. Eine Zuordnungsbeschreibung beispielsweise nach (5.7) und (5.8) bzw. (5.10) und (5.11) stellt lediglich die auf dem Bus abgelegten Daten, d. h. die Telegramme in Relation. Aus der Multiplikation von (5.7) mit seiner Transponierten und der anschließenden elementweisen Multiplikation ($.*$) mit der Einheitsmatrix $\underline{\underline{\boldsymbol{I}}}$, lässt sich jedem erzeugten Telegramm genau eine Kommunikationsfunktion zuweisen:

$$\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{K}}^{+} = \left(\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G}}^{+}\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G}}^{+\mathrm{T}}\right).* \underline{\underline{\boldsymbol{I}}}. \tag{5.9}$$

Ergänzend stellen folgende Quelle-Senke-Matrizen die Assoziationsattribute der Verbindungsklasse zwischen der logischen Topologie und der gerätebezogenen Datenvernetzung dar (vgl. Abbildung 5.1).

$$\underline{\underline{\boldsymbol{D}}}_{\boldsymbol{B}}^{+} = \left(d_{ij}\right) \text{ mit } d_{ij} = \begin{cases} 1, & \text{wenn Datum } i \text{ von Gerät } j \text{ via Bus versendet wird} \\ 0, & \text{sonst} \end{cases} = \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G}}^{+}\,\underline{\underline{\boldsymbol{G}}} \tag{5.10}$$

und

$$\underline{\underline{\boldsymbol{D}}}_{\boldsymbol{B}}^{-} = \left(d_{ij}\right) \text{ mit } d_{ij} = \begin{cases} 1, & \text{wenn Datum } i \text{ von Gerät } j \text{ via Bus empfangen wird} \\ 0, & \text{sonst} \end{cases} = \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G}}^{-}\,\underline{\underline{\boldsymbol{G}}}. \tag{5.11}$$

Im Vergleich zu den Matrizen (5.5) und (5.6) repräsentieren (5.10) und (5.11) ausschließlich die Relationen zwischen den Geräten und den auf den Bus zur Verfügung

stehenden Telegrammen. Die in diesem Abschnitt eingeführten strukturbeschreibenden Matrizen lassen sich für eine Erstellung bzw. automatische Generierung verschiedener weiterführender Analysemodelle nutzen. Beispielsweise wird im Abschnitt 5.1.2 das Busverhalten bei stochastisch angenommenen Telegrammversand untersucht. Die Abbildung der Kommunikationsfunktionen erfolgt über die Matrix $\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{K}}^{+}$ nach (5.9). Die ressourcenberücksichtigenden Telegrammübergaben auf dem Bus werden durch die Verknüpfung der Kommunikationsfunktionen zu den Transceivermodellen beschreibbar. Diese Verknüpfungen können der gerätebezogenen Datenvernetzung $\underline{\underline{\boldsymbol{D}}}_{\boldsymbol{B}}^{+}$ aus (5.10) entnommen werden (Abbildung 5.2).

5.1.2 Exemplarisches Simulationsmodell

Aus den im Systemmodell hinterlegten Informationen (Abbildung 5.1) lässt sich ein Petrinetz-basiertes Simulationsmodell ableiten, welches exemplarisch in Abbildung 5.2 dargestellt ist. Zunächst wird die Kommunikationsstruktur

$$\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{K}}^{+} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{matrix} \\ \\ \\ \\ \\ \text{Telegramme} \end{matrix}$$

$$\text{Kommunikationsfunktionen}$$

über die Formel (5.9) ermittelt und modelliert (Abbildung 5.2, links). Die Kommunikationsfunktionen $Kom1$ bis $Kom6$ werden hinsichtlich ihrer Prioritäten $\underline{\boldsymbol{o}}$ und Versandarten $\underline{\underline{\boldsymbol{A}}}_{\boldsymbol{K}}$ über die Matrizen (5.2) und (5.3) parametriert und dienen so der Stimulation des ressourcenorientierten Modells des Bussystems (CSMA/CA) aus Abbildung 5.2, rechts. Das Ressourcenmodell bildet innerhalb eines jeden Gerätetransceivers den Prozess für die Telegrammübergabe auf den Bus ab, d. h. den synchronisierten Buszugriff sowie die Prioritätensteuerung. In [Schrom 2003: 50] wurde bereits eine geeignete Darstellung diese CSMA/CA-Modells mit Petrinetzen vorgestellt. Über die Koppelstruktur

$$\underline{\underline{\boldsymbol{D}}}_{\boldsymbol{B}}^{+} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{matrix} \\ \\ \\ \\ \\ \text{Telegramme} \end{matrix} ,$$

$$\text{Geräte}$$

die in Abbildung 5.2, mittig dargestellt ist, wird die Kommunikationsstruktur (Abbildung 5.2, links) an das CSMA/CA-Modell (Abbildung 5.2, rechts) angebunden. Sofern eine Arbitrierung gewonnen wird, wird das der Kommunikationsfunktion zugeordnete Telegramm $Tel1$ bis $Tel6$ als abgesendet markiert.

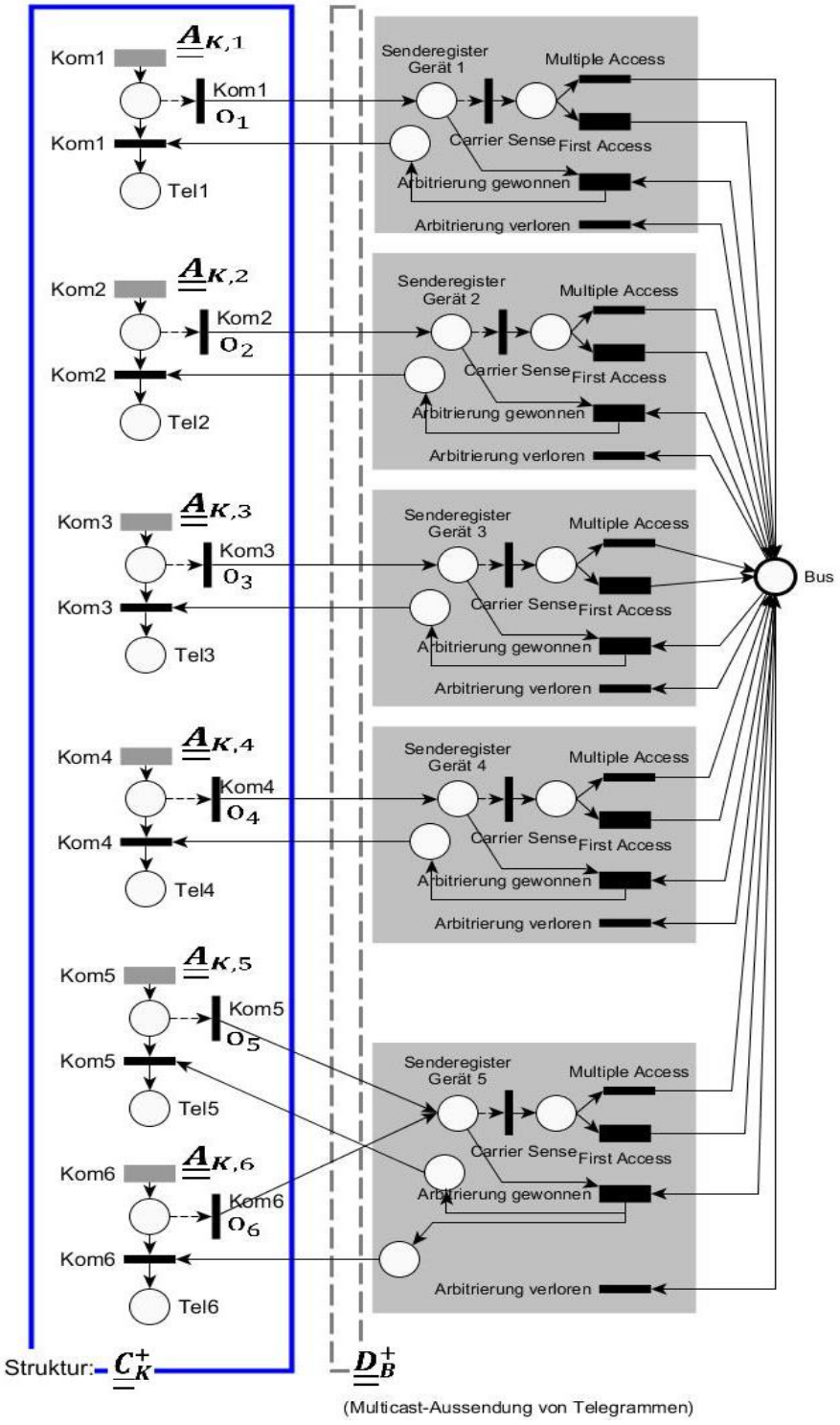


Abbildung 5.2: Abbildung des funktional betrachteten Kommunikationssystems (blau), gekoppelt an das Ressourcenmodell eines Bussystems bestehend aus fünf Gerätekomponenten am Beispiel von SmallCAN

5.1.3 Validierung

Das Beispiel aus Abbildung 5.2 zeigt sechs unabhängige Kommunikationsfunktionen auf der linken Seite, die in einem Bussystem mit fünf Gerätekomponenten (siehe rechte Seite) ausgeführt werden. Die Prioritäten der Kommunikationsfunktionen seien fest und wie folgt definiert:

$$\underline{\boldsymbol{o}} = \begin{pmatrix} o_1 \\ \vdots \\ o_6 \end{pmatrix} = \begin{pmatrix} 6 \\ 5 \\ 4 \\ 3 \\ 2 \\ 1 \end{pmatrix}.$$

D. h. dem Telegramm Tel1 wird die höchste und dem Telegramm Tel6 die niedrigste Priorität zugeordnet.

Zur Validierung ist das Modell mit verschiedenen Busauslastungen parametriert worden. Im Falle der Nachbildung einer maximalen Busauslastung ist das Sendeverhalten der Kommunikationsfunktionen entsprechend hoch gewählt worden:

$$\underline{\underline{\boldsymbol{A}}}_{\boldsymbol{K},100\%} = \begin{array}{c} \begin{array}{ccc} \text{Verteilung} & \text{Parameter 1} & \text{Parameter 2} \\ & \text{(untere Grenze)} & \text{(obere Grenze)} \end{array} \\ \begin{pmatrix} gleichverteilt & 31\ \text{ms} & 43\ \text{ms} \\ gleichverteilt & 31\ \text{ms} & 43\ \text{ms} \\ gleichverteilt & 31\ \text{ms} & 43\ \text{ms} \\ gleichverteilt & 31\ \text{ms} & 43\ \text{ms} \\ gleichverteilt & 31\ \text{ms} & 43\ \text{ms} \\ gleichverteilt & 5\text{s} & 10\ \text{s} \end{pmatrix} \end{array} \begin{array}{l} \\ \\ \\ \\ \\ \\ \text{Kommunikations-} \\ \text{funktionen} \end{array}.$$

Für das System SmallCAN entspricht das nahezu zyklisch angenommene Sendeverhalten $\underline{\underline{\boldsymbol{A}}}_{\boldsymbol{K},100\%}$ einer Busauslastung von 100 % , d. h. einer Telegrammrate von ca. 136 Telegrammen/s [Schrom 2003: 131].

Durch eine Simulation und eine messtechnische Erfassung an einem Realaufbau wurden die Verzögerungszeiten des niederpriorsten Telegrammes Tel6 analysiert. In Abbildung 5.3 (links) sind die Verzögerungszeiten in Abhängigkeit der Busauslastung gezeigt. Bei steigender Busauslastung steigen die Verzögerungszeiten für das erfolgreiche Ablegen des niederprioren Telegrammes auf den Bus exponentiell an [vgl. Tan et al. 1990], [vgl. Yang Yang/Tak-Shing Peter Yum 2003]. Während bei einer Auslastung von 70 % die maximale Verzögerungszeit bei lediglich 500 ms liegt steigt sie bei einer Auslastung von 100 % um den Faktor 7 an und beträgt dort 3,5 s. Für eine maximale Busauslastung ist in der Abbildung rechts die Verteilung der Verzögerungszeit aufgetragen. Für den Aufbau aus Abbildung 5.2, d. h. für ein System

mit wenigen Gerätekomponenten, wobei die Kommunikationsfunktionen entsprechend hohe Senderaten aufweisen, ergibt sich eine typische lognormale Verteilung $V_{K\,Tel\,6}(t)$ [vgl. Antoniou et al. 2002]. Bei maximaler Auslastung liegt die mittlere Verzögerungszeit bei ca. 500 ms (Abbildung 5.3, rechts).

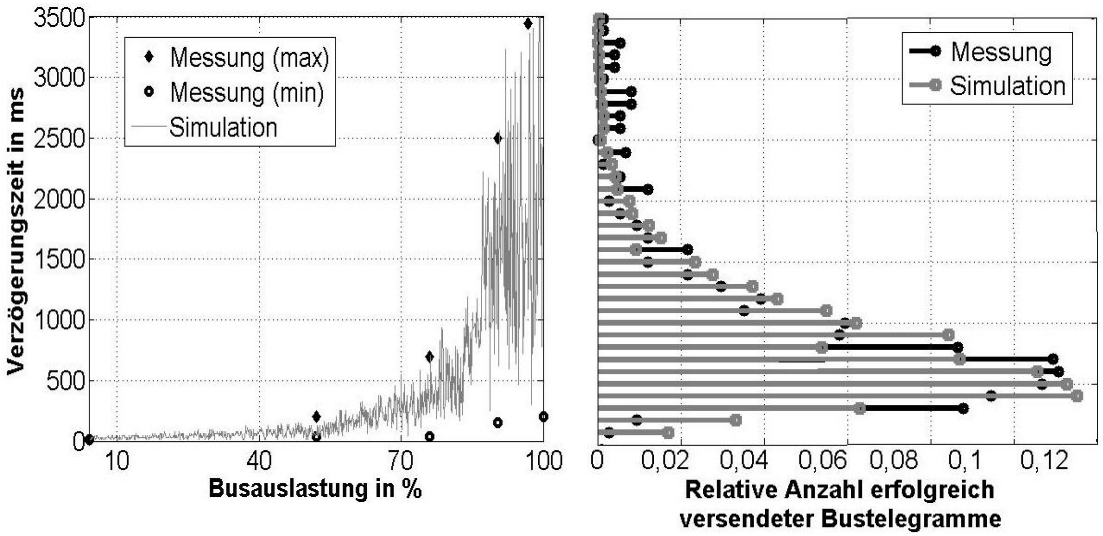


Abbildung 5.3: Verzögerungszeit eines niederprioren Telegrammes bei steigender Busauslastung von $\underline{\underline{A}}_{K,0\%}$ bis $\underline{\underline{A}}_{K,100\%}$ (links) und Verteilung der Verzögerungszeit bei maximaler Busauslastung $\underline{\underline{A}}_{K,100\%}$ (rechts)

5.1.4 Einfluss der Busauslastung

Die in Abbildung 5.4 dargestellten Simulationsergebnisse zeigen das Verzögerungsverhalten $V_K(t)$ beim Versand eines niederprioren Telegrammes bei verschiedenen Busauslastungen. Während bei geringer Busauslastung die Verzögerungszeiten noch klein sind und ein deterministisches Verhalten aufweisen (siehe Abbildung 5.4 $V_{K,10\%}(t)$), nähert sich die Verteilungsdichtefunktion bei erhöhter Auslastung immer mehr einer Lognormalverteilung an (Abbildung 5.4 $V_{K,100\%}(t)$). Die mittlere Verzögerungszeit steigt für höhere Busauslastungen deutlich an und beträgt bei maximaler Last ca. 500 ms (vgl. Abbildung 5.3 rechts), während sie bei einer Auslastung von 75 % lediglich bei 100 ms liegt. Weiterhin ist zu erkennen, dass bei einer Busauslastung von 50 % das erfolgreiche Absenden des niederpriorsten Telegrammes innerhalb von 300 ms garantiert werden kann. In Abbildung 5.4, unten sind zusammenfassend die Verläufe der Verzögerungszeiten logarithmisch aufgetragen.

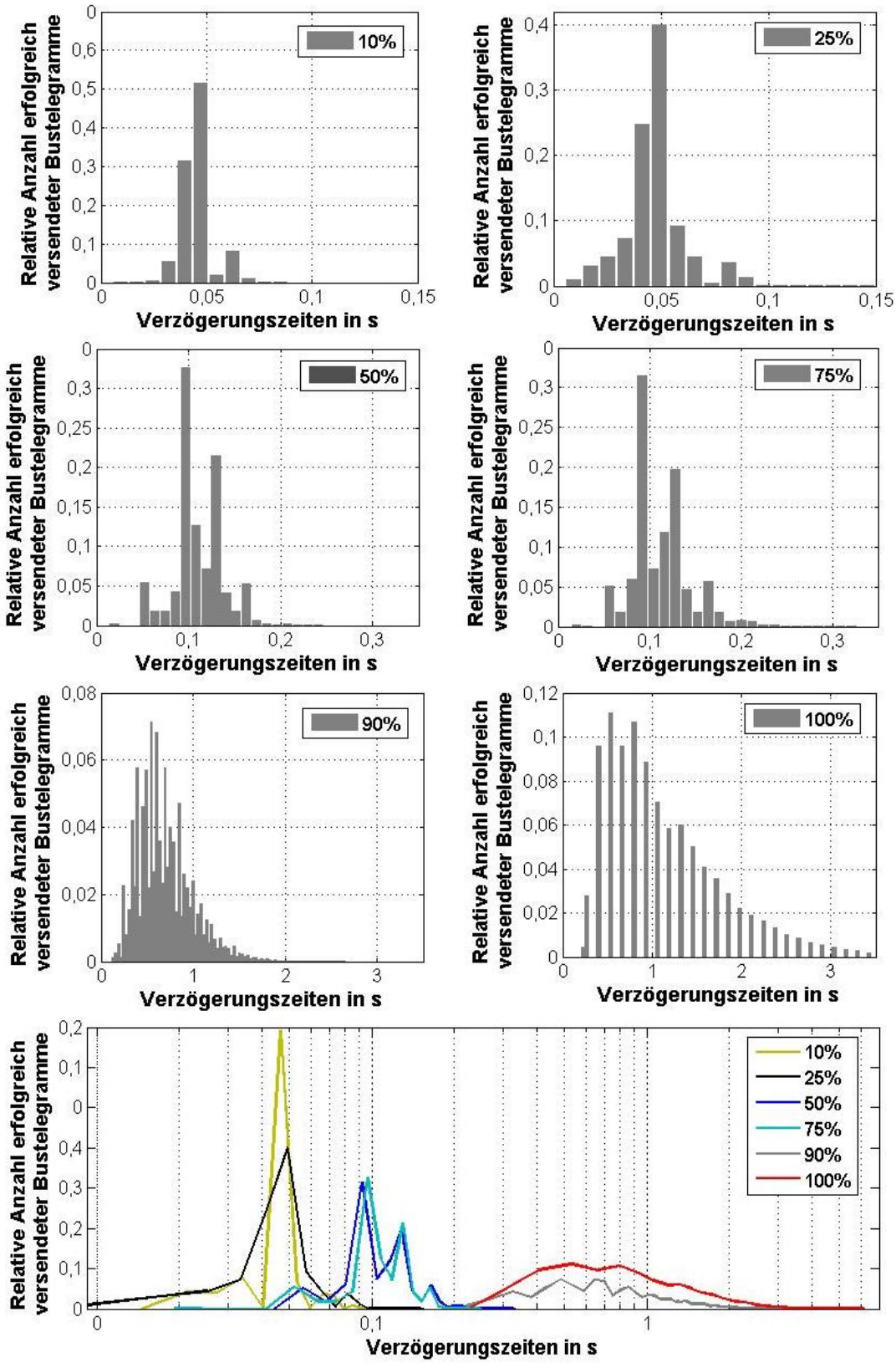


Abbildung 5.4: Simulierte Kommunikationsverzögerungen eines niederprioren Telegrammes bei unterschiedlichen Busauslastung

5.1.5 Einfluss der Versandart

Unter erhöhter Busauslastung wurde der Einfluss der Sendeart sowohl für niederpriore Telegramme als auch für Telegramme mittlerer Priorität untersucht. Aus Abbildung 5.5 (oben) ist zu erkennen, dass sich nach dem Systemstart für den Fall zyklisch gesendeter Botschaften diese in Nachrichtenfolgen regelmäßigen Abstandes eingliedern, d. h. es bilden sich definierte Zeitslots aus, in denen die Telegramme erfolgreich versandt werden. Das Phänomen verschwindet, sofern die Senderaten stochastischen Ursprungs sind (Abbildung 5.5, unten).

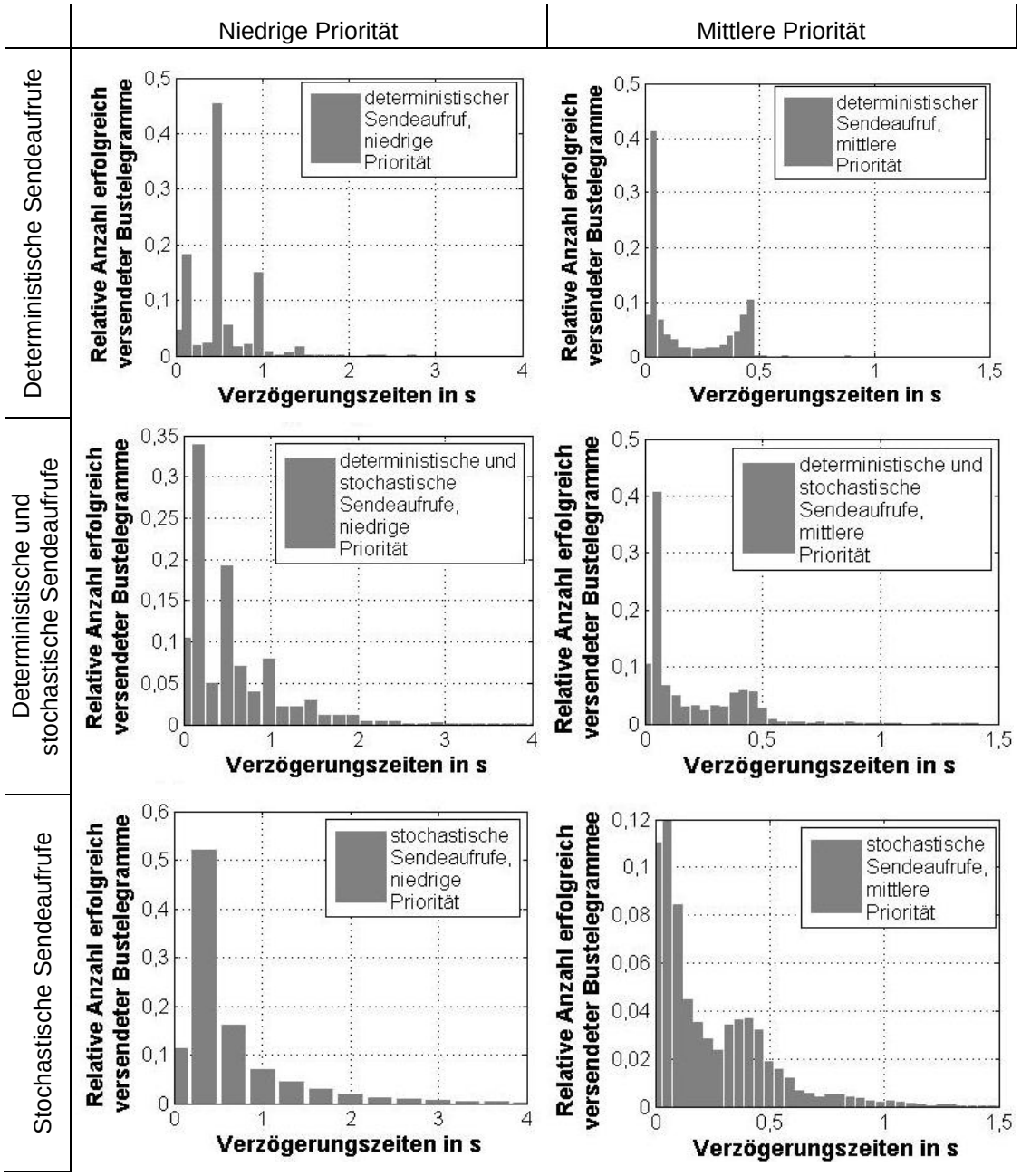


Abbildung 5.5: Einfluss der Versandart auf die Verzögerungszeiten von Telegrammen niedriger und mittlerer Priorität bei maximaler Busauslastung

5.1.6 Einfluss der Geräteanzahl

In Abbildung 5.6 wird der Einfluss einer steigenden Geräteanzahl auf die Verteilung der Kommunikationsverzögerung eines niederprioren Telegrammes bei zyklischer und bei ereignisbasierter Sendeart (deterministische und stochastische Aufrufe) aufgezeigt. Die Busauslastung wurde in allen Fällen konstant hoch gehalten. Es zeigen sich für eine steigende Geräteanzahl deutlich ausgeprägtere Nachrichtenfolgen konstanten Abstandes, insbesondere wenn die Botschaften zyklisch ausgesendet werden. Grund dafür ist eine gegenseitige Einstufung und Eingliederung der Telegramme.

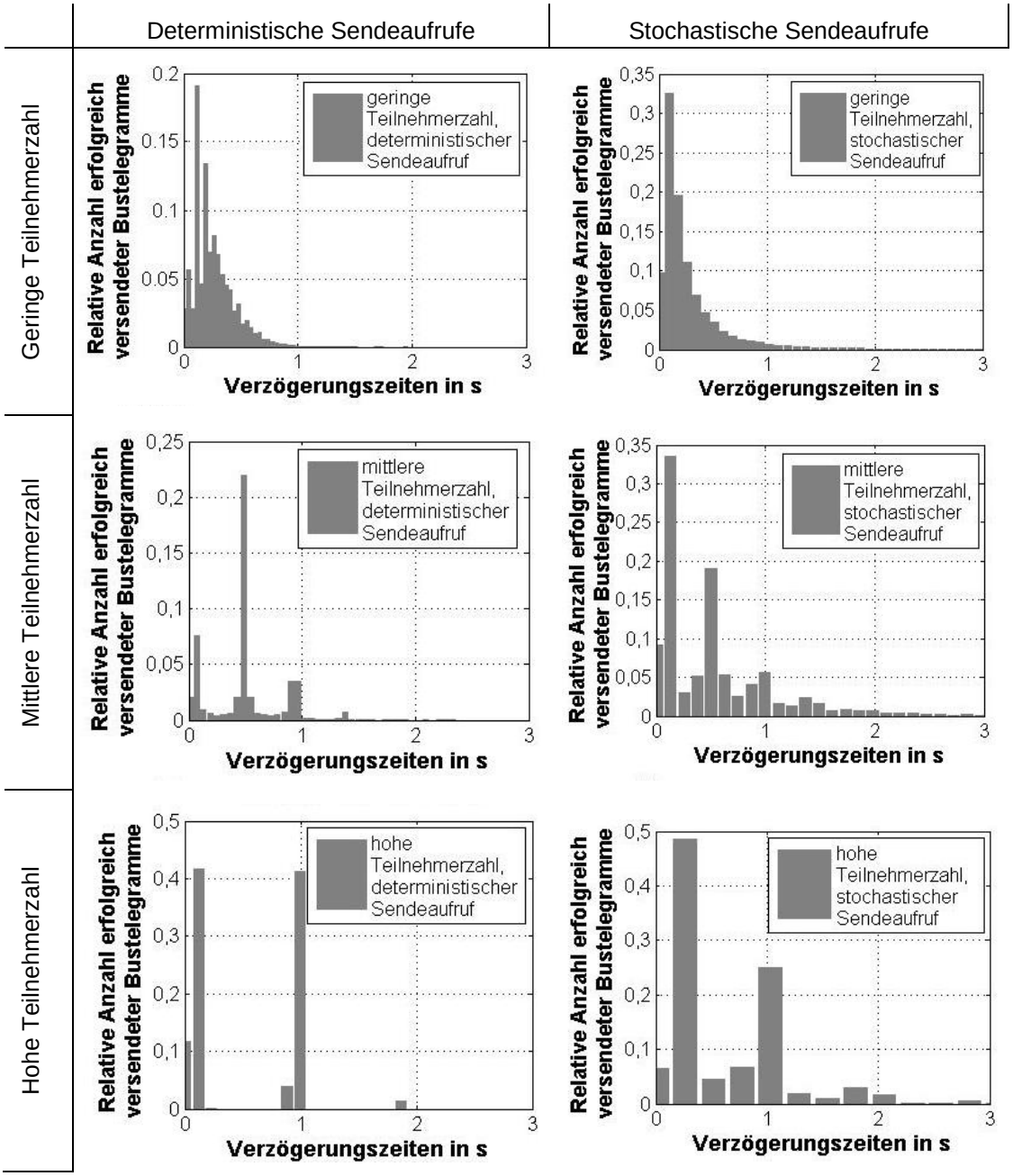


Abbildung 5.6: Einfluss der Teilnehmeranzahl auf die Verzögerungszeiten bei maximaler Busauslastung bei deterministischer und stochastischer Versandart

5.2 Zyklusübergangs- und Synchronisationsverhalten

Aufgrund der modularen Umsetzung einzelner Funktionen in zyklisch arbeitenden Programmkomponenten treten bei der Ausführung eines sequentiellen Ablaufs mehrerer verarbeitender Funktionen Zeitverzögerungen an den Übergängen der einzelnen Programmkomponenten auf. Denn an diesen muss auf den nächsten Zyklusbeginn des Folgeprogrammes gewartet werden, wodurch es zu Verzögerungen und damit zu größeren Laufzeiten verketteter Funktionsabläufe kommen kann [Schnieder 1993]. In diesem Abschnitt wird auf die entstehenden Zeitschwankungen an Zyklus- und Synchronisationsübergängen zwischen Programmkomponenten eingegangen. In den folgenden Teilabschnitten werden das dieser Problemstellung zuzuordnende Systemmodell (Abschnitt 5.2.1) sowie die daraus ableitbaren Simulationsmodelle (Abschnitt 5.2.2) vorgestellt. Anschließend sind in Abschnitt 5.2.3 die Validierung des Simulationsmodells und in Abschnitt 5.2.4 die Vorstellung der Simulationsergebnisse für einen parallelen Verbund behandelt.

5.2.1 Systemmodell

Ein geeignetes Systemmodel ist in Abbildung 5.7 gezeigt. Es umfasst u. a. die Begriffshierarchie des *Anwendungssystems* (Abbildung 5.7, violett) aus funktionaler und technologischer Sicht, d. h. zur Beschreibung dieses Modells sind neben den zu koppelnden Funktionen auch die Übergangsprozesse zwischen den einzelnen Programmkomponenten berücksichtigt. Weiter wird das *Programmsystem* als Ressource der Systemfunktion unter räumlichen und technologischen Aspekten beschrieben.

Die *Informationsverarbeitung*, d. h. die Systemfunktion des verteilten Anwendersystems, wird durch einzelne Verarbeitungsfunktionen umgesetzt. Für diese können zeitliche Angaben in Form des folgenden Vektors mit als konstant angenommenen Bearbeitungszeiten gemacht werden:

$$\underline{\boldsymbol{t_b}} = \left(t_{b_i}\right) \text{ mit } t_{b_i} = \text{ Bearbeitungszeit der Funktion } i. \tag{5.12}$$

Weiterhin lässt sich das *Aufrufverhalten* der verarbeitenden Funktionen über folgende Matrix mit hinterlegten Verteilungsfunktionen *dist* und deren Parameter *par* angeben:

$$\underline{\boldsymbol{A_F}} = (a_i) = \begin{matrix} \text{Verteilung} & \text{Parameter 1} & \text{Parameter 2} \\ \begin{pmatrix} dist_1 & par1_1 & \dots \\ \vdots & \vdots & \\ dist_i & par1_i & \dots \end{pmatrix} & \begin{matrix} \text{verarbeitende-} \\ \text{Funktion} \end{matrix} \end{matrix}. \tag{5.13}$$

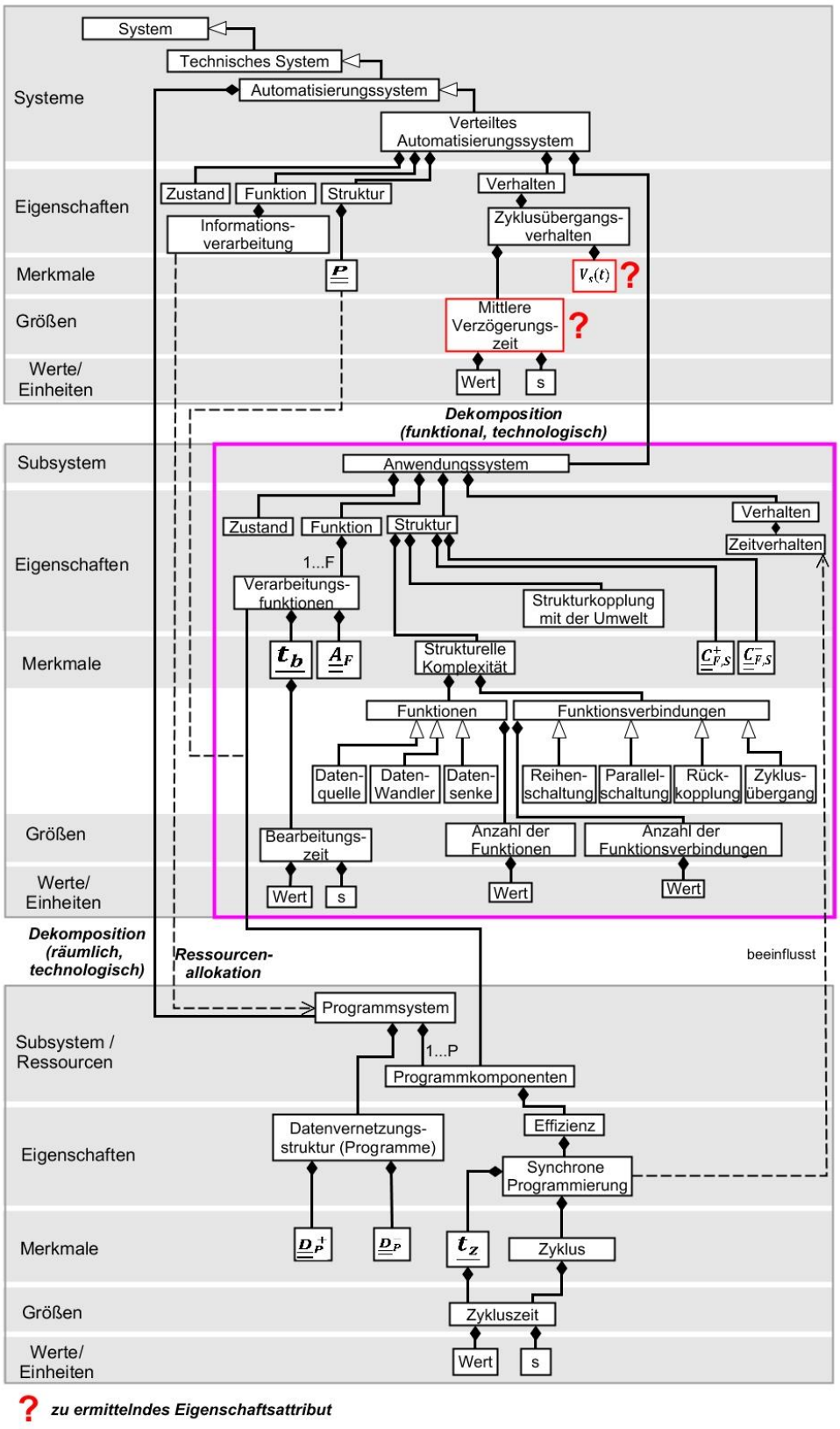


Abbildung 5.7: Ganzheitliches Systemmodell mit funktional und technologisch zu betrachteten Anwendungssystem (violett) angepasst an die Problemstellung des Zyklusübergangs- und Synchronisationsverhalten in einem verteilten Automatisierungssystem

Die Zuordnung der Verarbeitungsfunktionen auf die einzelnen *Programmkomponenten* lässt sich über das Assoziationsattribut der Programmmatrix $\underline{\underline{\boldsymbol{P}}}$ beschreiben:

$$\underline{\underline{\boldsymbol{P}}} = (p_{ij}) \text{ mit } p_{ij} = \begin{cases} 1, & \text{wenn Funktion } i \text{ auf Programmkomponente } j \text{ realisiert ist} \\ 0, & \text{sonst} \end{cases}. \tag{5.14}$$

Ergänzend können zu den Programmkomponenten Angaben zu deren Zykluszeiten gemacht werden, für die der folgende *Zykluszeitenvektor* eingeführt wird:

$$\underline{\boldsymbol{t_z}} = (t_{z_i}) \text{ mit } t_{z_i} = \text{Zykluszeit der Programmkomponente } i. \tag{5.15}$$

Die Realisierung der *Informationsverarbeitung* erfolgt durch sämtliche Programmkomponenten, d. h. durch das *Programmsystem*. Die Zuordnung der Programmkomponenten zu den von ihnen erzeugten und empfangenen Daten lässt sich über die folgenden Matrizen der *Datenvernetzung* im Programmsystem angeben:

$$\underline{\underline{\boldsymbol{D}}}_{\boldsymbol{P}}^{+} = (d_{ij}) \text{ mit } d_{ij} = \begin{cases} 1, & \text{wenn Datum } i \text{ von Programmkomponente } j \text{ versendet wird} \\ 0, & \text{sonst} \end{cases}$$

$$= \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F}}^{+} \, \underline{\underline{\boldsymbol{P}}} \tag{5.16}$$

und

$$\underline{\underline{\boldsymbol{D}}}_{\boldsymbol{P}}^{-} = (d_{ij}) \text{ mit } d_{ij} = \begin{cases} 1, & \text{wenn Datum } i \text{ von Programmkomponente } j \text{ empfangen wird} \\ 0, & \text{sonst} \end{cases}$$

$$= \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F}}^{-} \, \underline{\underline{\boldsymbol{P}}}. \tag{5.17}$$

Die Daten, die ausschließlich zwischen Programmkomponenten ausgetauscht werden, werden folglich als Nachrichten benannt. Aus dem *Nachrichtenübermittlungsvektor*

$$\underline{\boldsymbol{d}}_{\boldsymbol{P}} = (d_i) \text{ mit } d_i = \underline{\underline{\boldsymbol{D}}}_{\boldsymbol{P},ij}^{-} \left(\underline{\mathbf{1}} - \underline{\boldsymbol{I}}_j\right) \wedge \underline{\underline{\boldsymbol{D}}}_{\boldsymbol{P},ij}^{+} = 1 \tag{5.18}$$

ergeben sich die *Programmübergangsmatrizen*

$$\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{P}}^{+} = (c_{ij}) \text{ mit } c_{ij} = \begin{cases} 1, & \text{wenn } \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F},ij}^{+}=1 \ \wedge \underline{\boldsymbol{d}}_{\boldsymbol{P},i}\neq 0 \\ 0, & \text{sonst} \end{cases} \tag{5.19}$$

und

$$\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{P}}^{-} = (c_{ij}) \text{ mit } c_{ij} = \begin{cases} 1, & \text{wenn } \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F},ij}^{-}=1 \ \wedge \underline{\boldsymbol{d}}_{\boldsymbol{P},i}\neq 0 \\ 0, & \text{sonst} \end{cases}. \tag{5.20}$$

5.2.2 Simulationsmodell

Zur Erstellung eines geeigneten Simulationsmodells in Form eines Petrinetzes lassen sich die Inzidenzmatrizen des funktionalen Anwendersystems $\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F}}^{+}$ und $\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F}}^{-}$ aufspalten und mit den Programmübergangsmatrizen aus (5.19) und (5.20) wie folgt verknüpfen:

$$\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F,S}}^{+} = \begin{matrix} \text{verarbeitende Funktionen} & \text{Zyklusübergänge} \\ \begin{pmatrix} \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{11}} & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{22}} \end{pmatrix} & \begin{matrix} \text{Daten} \\ \text{Nachrichten} \end{matrix} \end{matrix}$$

$$= \begin{pmatrix} \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F}}^{+} & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{\mathbf{0}}} & \left(\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{P}}^{+}\, \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{P}}^{+\mathrm{T}}\right) .* \underline{\underline{\boldsymbol{I}}} \end{pmatrix} \tag{5.21}$$

und

$$\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F,S}}^{-} = \begin{matrix} \text{verarbeitende Funktionen} & \text{Zyklusübergänge} \\ \begin{pmatrix} \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{11}} & \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{12}} \\ \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{21}} & \underline{\underline{\mathbf{0}}} \end{pmatrix} & \begin{matrix} \text{Daten} \\ \text{Nachrichten} \end{matrix} \end{matrix}$$

$$= \begin{pmatrix} \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F}}^{-} - \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{P}}^{-} & \left(\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{P}}^{-}\, \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{P}}^{-\mathrm{T}}\right) .* \underline{\underline{\boldsymbol{I}}} \\ \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{P}}^{-} & \underline{\underline{\mathbf{0}}} \end{pmatrix}. \tag{5.22}$$

Demnach wird das rein funktional betrachtete Modell des Anwendungssystems gemäß Abbildung 3.11, rot oder Abbildung 5.1, rot um technologisch bedingte Zyklusübergänge erweitert. Die Matrizen (5.21) und (5.22) stellen damit die Struktur desjenigen Teilsystems dar, welches zur Untersuchung der Zyklusübergangs- und Synchronisationsprozesse heranzuziehen ist. In dem Modell wird zunächst von der Beeinflussung der Verzögerungen durch die Buskommunikation, die im vorherigen Abschnitt behandelt wurde, abstrahiert.

Die Struktur des Teilsystems gemäß (5.21) und (5.22) ist in Abbildung 5.8 auf der linken Seite exemplarisch für einen linearen Verbund von acht verarbeitenden Funktionen dargestellt.

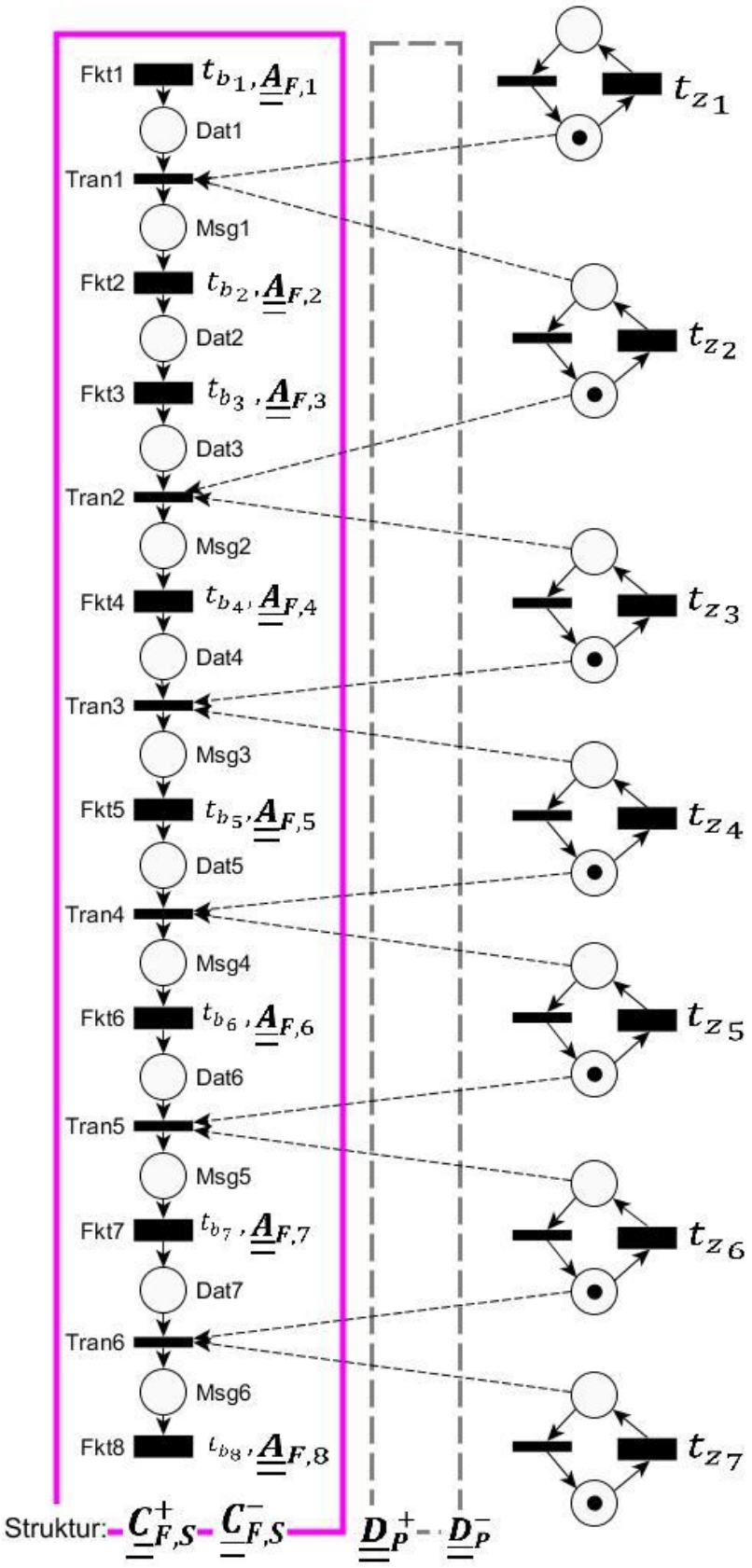


Abbildung 5.8: Petrinetzmodell eines linear verknüpften Verbundes mit sieben Programmkomponenten

Die Struktur bildet neben den verarbeitenden Funktionen Fkt und den Daten Dat auch die Zyklusübergänge $Tran$ und die Nachrichten Msg ab, d. h. diejenigen Daten aus dem funktional betrachteten Anwendersystem, die zwischen den Programmkomponenten ausgetauscht werden. In dem Modell sind auf der linken Seite die im Systemmodell hinterlegten Strukturmerkmale $\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{F,S}}$, $\underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{F,S}}$ sowie die Kenngrößen $\underline{\underline{\boldsymbol{A}}}_{\boldsymbol{F}}$ und $\underline{\mathbf{t}}_{\boldsymbol{B}}$ als Parameter der behafteten Transitionen, d. h. den verarbeitenden Funktionen, wiederzufinden. Die rechte Seite des Netzes zeigt in der Ressourcendarstellung die Programmkomponenten mit hinterlegten Zykluszeiten $\underline{\mathbf{t}}_{\boldsymbol{z}}$ gemäß den Angaben aus dem Systemmodell aus Abbildung 5.7. Durch die strukturelle Kopplung $\underline{\underline{\boldsymbol{D}}}^{+}_{\boldsymbol{P}}$ und $\underline{\underline{\boldsymbol{D}}}^{-}_{\boldsymbol{P}}$ (Abbildung 5.8, mittig), die über die eingeführten Matrizen (5.16) und (5.17) beschrieben ist, werden die Prozessschritte zur Nachrichtenübergabe $Tran1$ bis $Tran6$ in Abhängigkeit des Zyklusverhaltens der Programmkomponenten aktivierbar.

5.2.3 Validierung

Im Folgenden wird die Validierung des Petrinetzmodells aus Abbildung 5.8 basierend auf einen Vergleich simulierter, analytischer und messtechnischer Ergebnisse beschrieben. Die Darstellung zeigt auf der linken Seite exemplarisch den sequentiellen Ablauf von acht verarbeitenden Funktionen unter Berücksichtigung von sechs Programmübergängen. Generell ist das Schaltverhalten der Transitionen $Fkt1$ bis $Fkt8$ abhängig von den Bearbeitungszeiten und dem Aufrufverhalten der verarbeitenden Funktionen. Während für $Fkt1$ ein gleichverteilter Funktionsaufruf $\underline{\underline{\boldsymbol{A}}}_{\boldsymbol{F},1}$ definiert wird, liegt aufgrund des direkten kausalen Zusammenhanges zwischen den linear gekoppelten Funktionen für die Funktionsaufrufverhalten $\underline{\underline{\boldsymbol{A}}}_{\boldsymbol{F},2}$ bis $\underline{\underline{\boldsymbol{A}}}_{\boldsymbol{F},8}$ eine folgegesteuerte Aktivierung vor:

$$\underline{\underline{\boldsymbol{A}}}_{\boldsymbol{F}} = \begin{array}{c} \begin{array}{ccc} \text{Verteilung} & \text{Parameter 1} & \text{Parameter 2} \\ & \text{(untere Grenze)} & \text{(obere Grenze)} \end{array} \\ \begin{pmatrix} gleichverteilt & 10\ \text{s} & 20\ \text{s} \\ folgegesteuert & & \\ folgegesteuert & & \\ folgegesteuert & & \\ folgegesteuert & & \\ folgegesteuert & & \\ folgegesteuert & & \\ folgegesteuert & & \end{pmatrix} \end{array} \begin{array}{l} \\ \\ \\ \\ \\ \\ \\ \\ \text{verarbeitende-} \\ \text{Funktion} \end{array} .$$

Das Funktionsaufrufverhalten $\underline{\underline{A}}_{F}$ entspricht damit einem linear gekoppelten Funktionsverbund.

Die Bearbeitungszeiten (5.12) der Funktionen sind gegenüber den Zykluszeiten der Programmkomponenten vernachlässigbar klein gewählt worden ($t_b \ll t_z$), so dass im Wesentlichen das Zyklusübergangsverhalten $V_S(\mathrm{t})$ nachgebildet wird:

$$\underline{\boldsymbol{t_b}} = \begin{pmatrix} t_{b_1} \\ \vdots \\ t_{b_8} \end{pmatrix} = \begin{pmatrix} 10\ \mathrm{ms} \\ \vdots \\ 10\ \mathrm{ms} \end{pmatrix}.$$

Das Schaltenverhalten der Programmübergänge $Tran1$ bis $Tran6$ wird durch den Ablauf des Ressourcenmodells auf der rechten Seite bestimmt. Das Ressourcenmodell setzt sich aus den sieben Programmkomponenten zusammen, die an dem Funktionsablauf beteiligt sind. Deren Zykluszeiten seien gemäß (5.15) wie folgt definiert:

$$\underline{\mathbf{t}}_{\boldsymbol{z}} = \begin{pmatrix} t_{z_1} \\ \vdots \\ t_{z_7} \end{pmatrix} = \begin{pmatrix} 1\ \mathrm{s} \\ \frac{6}{5}\ \mathrm{s} \\ \frac{7}{5}\ \mathrm{s} \\ 1\ \mathrm{s} \\ \frac{6}{5}\ \mathrm{s} \\ \frac{7}{5}\ \mathrm{s} \\ 1\mathrm{s} \end{pmatrix}.$$

Die simulativ und messtechnisch ermittelten Verläufe der Verzögerungszeiten nach den jeweiligen Programmübergängen können Abbildung 5.9 entnommen werden. Die Verteilungsdichteverläufe der Verzögerungszeiten weisen für die Simulation und für die Messung am Realssystem nach allen Programmkomponenten gleiche Tendenzen auf und sind durch nahezu identische Mittel- und Extremwerte gekennzeichnet. Lediglich bei den Messungen sind neben den Verzögerungszeiten vernachlässigbar kleine aber nicht vermeidbare interne Bearbeitungszeiten mit erfasst worden. Gemäß dem *zentralen Grenzwertsatz* nähert sich die Verteilungsfunktion mit steigender Anzahl verketteter Programmkomponenten einer Normalverteilung an.

Ähnliche Ergebnisse konnten auch in [Höme/Dietrich 2012] am Beispiel eines PROFINET-Systems vorgestellt werden. Wie $V_{S\ Fkt1 \rightarrow Fkt2}(t)$ in dem Intervall $[0, 1\ \mathrm{s}]$ zeigt, kann das Zyklusübergangsverhalten zwischen zwei Programmkomponenten als gleichverteilt angenommen werden (Abbildung 5.9, oben links), [Schnieder 1993]. Die maximalen Verzögerungszeiten nach den jeweiligen Programmübergängen entsprechen den analytisch bestimmbaren Kenngrößen gemäß [Schnieder 1993]:

$$\Delta t_{max,i} = \sum_{i=1}^{P} t_{z_i} - \max(t_{z_i}) \ ,$$

$$\Delta T_{max} = \{\Delta t_{max,1}\ , \dots\ ,\ \Delta t_{max,P}\} = \{0\text{s},\ 1\text{s},\ \tfrac{11}{5}\text{s},\ \ \tfrac{16}{5}\text{s},\ \ \tfrac{23}{5}\text{s},\ \ \tfrac{28}{5}\text{s},\ \ \tfrac{34}{5}\text{s}\}\ . \qquad (5.23)$$

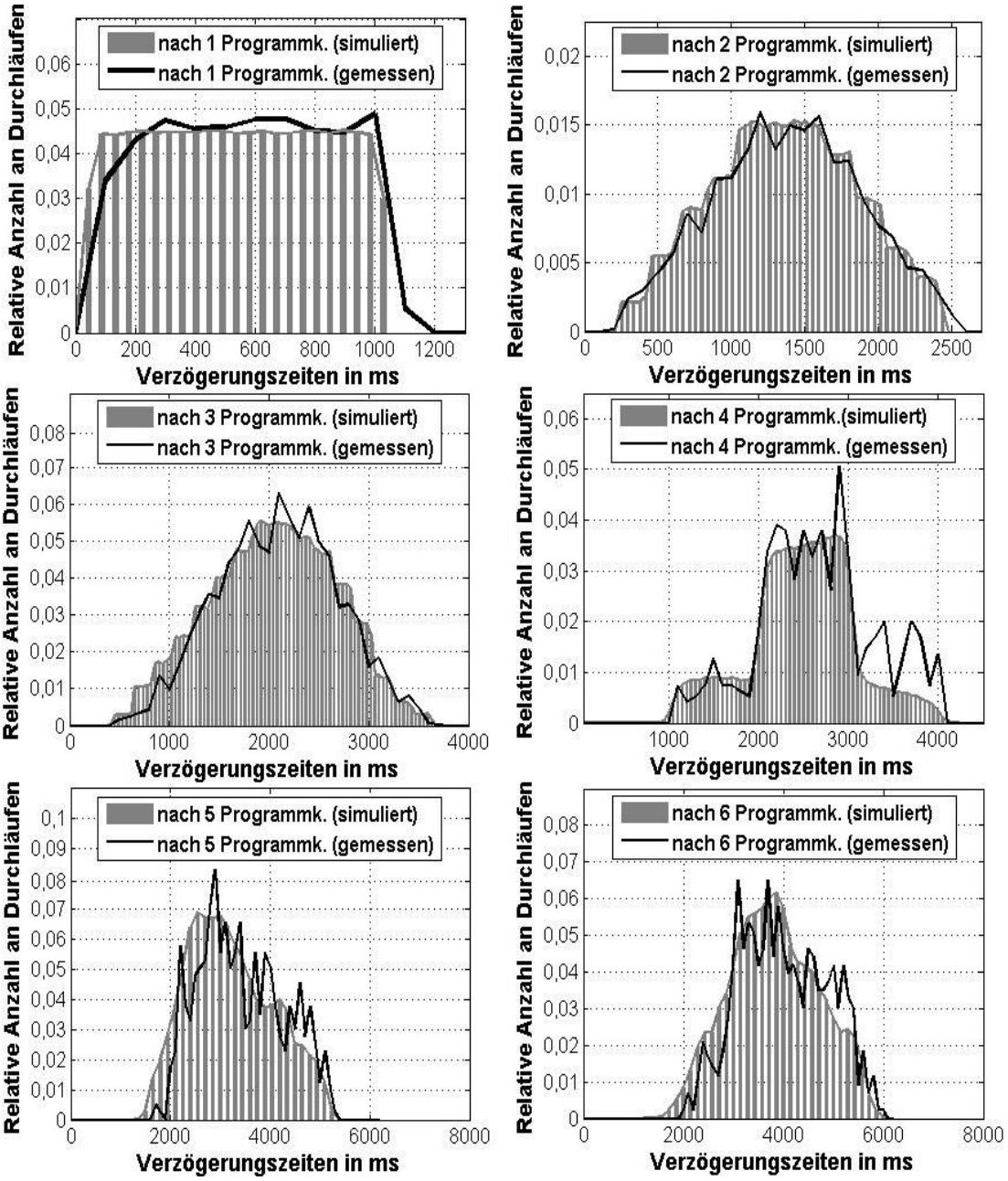


Abbildung 5.9: Verzögerungsverhalten an den Programmübergängen eines linearen Verbundes mit sieben Programmkomponenten verschiedener Zykluszeiten

Neben den erzeugten Verteilungen (Abbildung 5.9) werden in Abbildung 5.10 die zeitlichen Verläufe der Verzögerungen für einen linearen Verbund dargestellt. Gemäß den Ergebnissen aus [Schnieder 1993] und [Schnieder/Kraft 1980] weist der zeitliche Verlauf nach den ersten Zyklusübergängen eine periodische Folge auf. Nach der Kopplung weiterer Programmkomponenten geht das Verzögerungsverhalten in eine

Normalverteilung über und eine periodische Abfolge im zeitlichen Verlauf ist nicht mehr erkennbar.

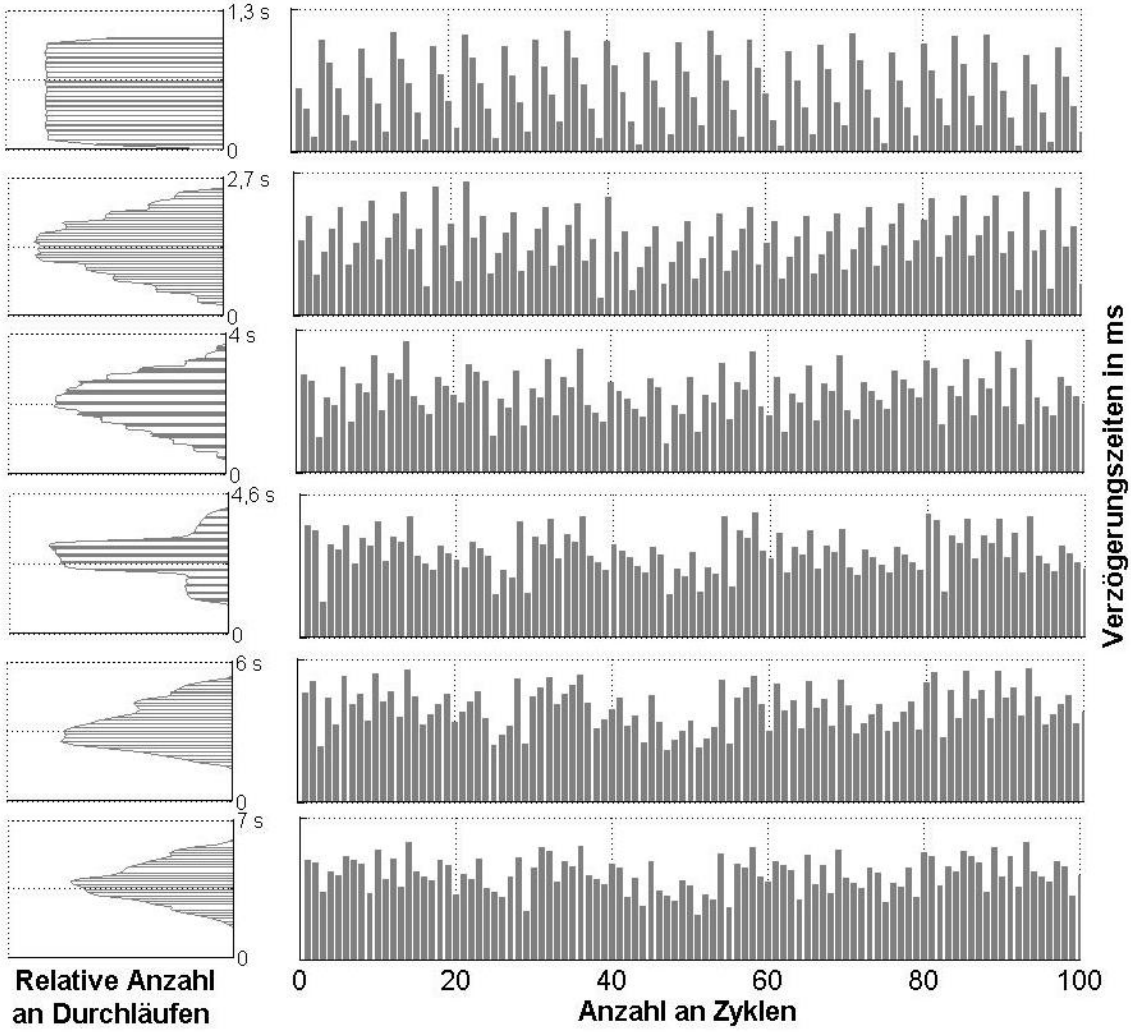


Abbildung 5.10: Simulierter zeitliche Verlauf der Verzögerungszeiten an den Programmübergängen eines linearen Verbundes mit sieben Programmkomponenten verschiedener Zykluszeiten

5.2.4 Simulation eines parallelen Verbundes

Auf der Basis des validierten Petrinetzmodells werden im Folgenden ausgewählte Simulationsergebnisse zur Ermittlung des Übergangsverhaltens eines parallelen Verbundes behandelt. Die parallele Anordnung der Programmkomponenten 1 bis 6 ist in Abbildung 5.11 dargestellt. Eine abschließende Datensynchronisation ist durch die Funktion $Fkt8$ erfolgt, die auf der Programmkomponente 7 abgelegt ist.

Zur Simulation der parallel angeordneten Programmkomponenten sind folgende Funktionsaufrufverhalten gewählt worden:

$$\underline{\underline{\boldsymbol{A}}}_{\boldsymbol{F},det} = \begin{pmatrix} \text{Verteilung} & \text{Parameter 1} \\ deterministisch & 0{,}5\ \text{s} \\ deterministisch & 0{,}5\ \text{s} \\ deterministisch & 0{,}5\ \text{s} \\ deterministisch & 0{,}5\ \text{s} \\ deterministisch & 0{,}5\ \text{s} \\ deterministisch & 0{,}5\ \text{s} \\ deterministisch & 0{,}5\ \text{s} \\ folgegesteuert & \end{pmatrix} \begin{matrix} \\ \\ \\ \\ \\ \\ \\ \\ \end{matrix} \begin{matrix} \\ \text{verarbeitende-} \\ \text{Funktion} \end{matrix}$$

bzw.

$$\underline{\underline{\boldsymbol{A}}}_{\boldsymbol{F},uni} = \begin{pmatrix} \text{Verteilung} & \text{Parameter 1} & \text{Parameter2} \\ & \text{(untere Grenze)} & \text{(obere Grenze)} \\ gleichverteilt & 0\ \text{s} & 1\ \text{s} \\ gleichverteilt & 0\ \text{s} & 1\ \text{s} \\ gleichverteilt & 0\ \text{s} & 1\ \text{s} \\ gleichvereilt & 0\ \text{s} & 1\text{s} \\ gleichverteilt & 0\ \text{s} & 1\text{s} \\ gleichverteilt & 0\ \text{s} & 1\text{s} \\ gleichverteilt & 0\ \text{s} & 1\text{s} \\ folgegesteuert & & \end{pmatrix} \begin{matrix} \\ \\ \\ \\ \\ \\ \\ \\ \end{matrix} \begin{matrix} \\ \text{verarbeitende-} \\ \text{Funktion} \end{matrix} .$$

Die angegebenen Verhalten entsprechen einem zyklischen bzw. gleichverteilten Aufruf der verarbeitenden Funktionen $Fkt1$ bis $Fkt7$, wobei entsprechend der Struktur aus Abbildung 5.11 für die $Fkt3$ eingangsseitig zuvor das Datum $Dat2$ erwartet wird. Die Funktion $Fkt8$ wird folgegesteuert durch die Nachrichten $Msg1$ bis $Msg6$ aktiviert. Die Bearbeitungszeiten der verarbeitenden Funktionen sind weiterhin als vernachlässigbar klein anzunehmen.

In Abbildung 5.12 und Abbildung 5.13 ist das Verzögerungsverhalten $V_s(t)$ durch die Synchronisation paralleler Prozesse an $Fkt8$ für ein deterministisches und stochastisches Aufrufverhalten der Funktionen gezeigt. Die Anzahl der zu synchronisierenden parallelen Prozesse wird dabei schrittweise erhöht. Im Falle einer Parallelschaltung der Programmkomponenten 1 und 2 ist, bedingt durch die in Reihe geschalteten Funktionen $Fkt2$ und $Fkt3$ innerhalb der Programmkomponenten 2, insbesondere bei dem deterministischen Aufrufverhalten $\underline{\underline{\boldsymbol{A}}}_{\boldsymbol{F},det}$ eine deutliche Stufe in der Verteilungsfunktion $V_{S_{(Fkt1 \ldots Fkt3 \rightarrow Fkt8)}}(t)$ erkennbar.

In Abbildung 5.12 sind im abfallenden Verlauf der Verteilungsfunktion kleinere Plateaus sichtbar, die auf die unterschiedlichen Zykluszeiten weiterer parallel gekoppelter Programmkomponenten zurückzuführen sind. Insgesamt wird ersichtlich, dass der Verlauf mit steigender Anzahl zu synchronisierender Prozesse in Richtung des Mittelwertes stärker ansteigt.

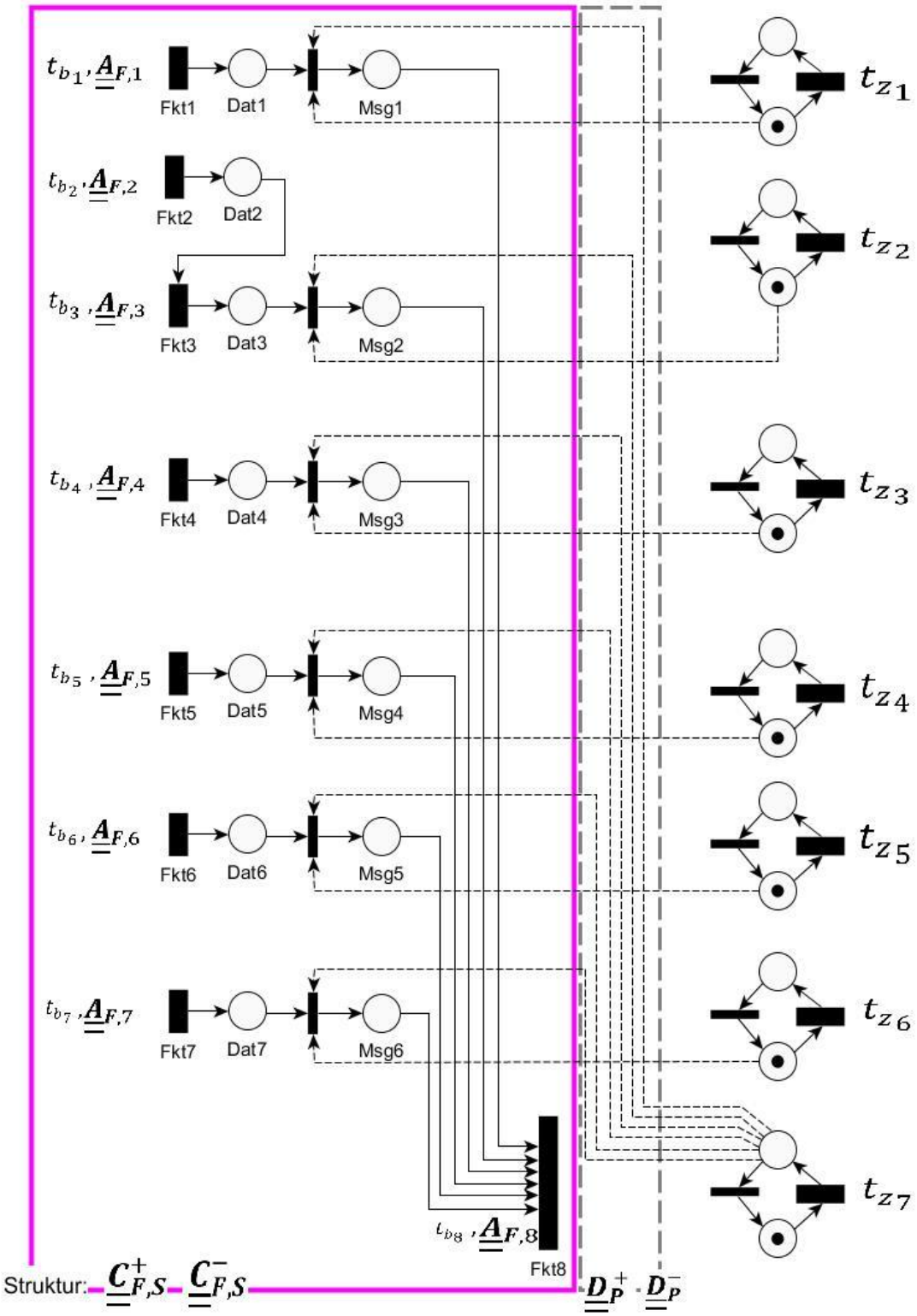


Abbildung 5.11: Petrinetzmodell eines parallelen Verbundes mit sechs Programmkomponenten und deren Synchronisation an der siebten Programmkomponente

Im Falle des Aufbaus nach Abbildung 5.11 und unter der Annahme streng deterministischer Funktionsaufrufe $\underline{\underline{\boldsymbol{A}}}_{\boldsymbol{F},det}$ errechnet sich der Mittelwert der Verzögerungszeit aus der Addition der Zeit für den längsten Funktionsablauf innerhalb des Parallelverbundes (hier: $Fkt2 \rightarrow Fkt3$) und der mittleren Zyklusübergangszeit zwischen dem Parallelverbund, bestehend aus den Programmkomponenten und der Synchronisationsfunktion $Fkt8$ zu

$$\bar{t}_{V_{S_{(Fkt1 \ldots Fkt7 \rightarrow Fkt8)}}} = par1_2 + par1_3 + \frac{(\max(t_{z_1} \ldots t_{z_6}) + t_{z_7}) - \max(\max(t_{z_1} \ldots t_{z_6}), t_{z_7})}{2} = 1{,}5 \text{ s}.$$

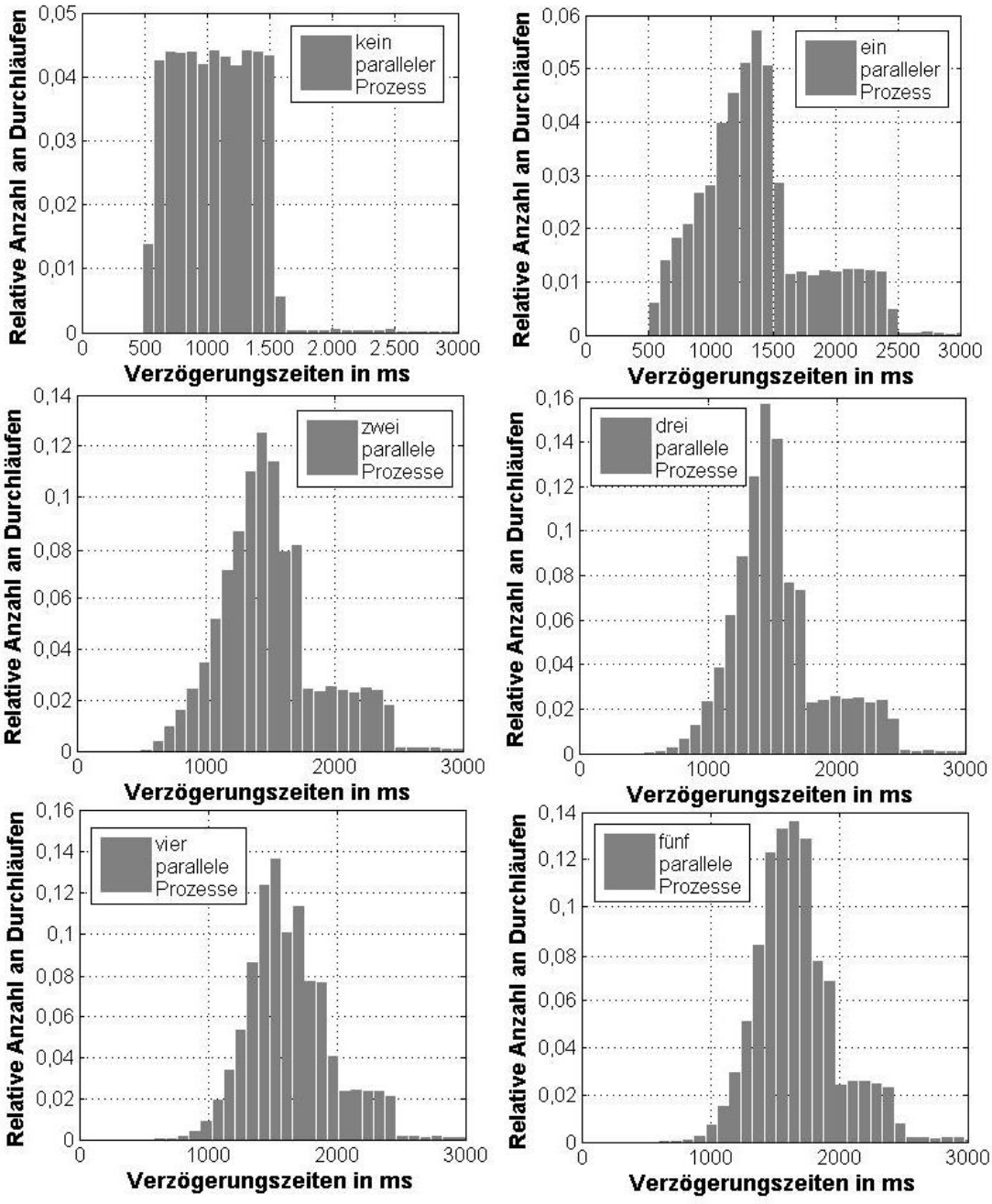


Abbildung 5.12: Verzögerungsverhalten durch die Synchronisation eines parallelen Verbundes bei deterministischen Funktionsaufrufen

Für ein gleichverteiltes Aufrufverhalten $\underline{\boldsymbol{A}}_{F,uni}$ stellt sich bei der Synchronisation vieler paralleler Abläufe zunehmend eine Normalverteilung ein. Im Vergleich zum deterministischen Aufrufverhalten verschiebt sich der Mittelwert der Verteilungsfunktion $V_s(t)$ um die Obergrenze der am häufigsten vorkommenden Aufrufverteilung weiter in Richtung höherer Verzögerungszeiten.

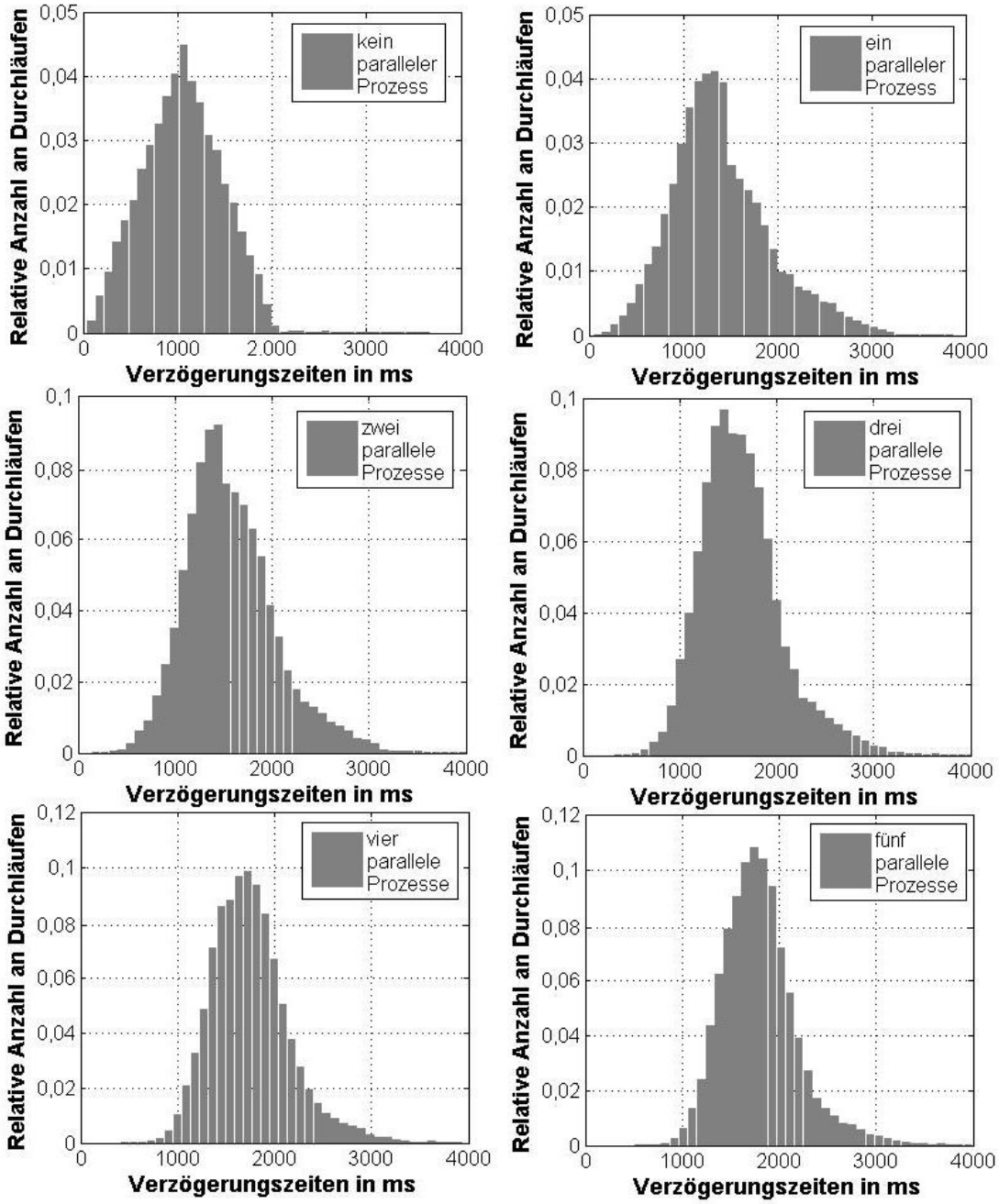


Abbildung 5.13: Verzögerungsverhalten durch die Synchronisation eines parallelen Verbundes bei stochastischen Funktionsaufrufen

5.3 Laufzeitermittlung eines kausalen Prozessablaufes

In Abschnitt 5.1 wurde das Kommunikationsverhalten gänzlich isoliert und allgemein, d. h. ohne Berücksichtigung anwendungsorientierter Funktionsaufrufe betrachtet. In Abschnitt 5.2 wurde eine erweiterte Darstellung des funktionalen Anwendungssystems um Zyklusübergänge bzw. Synchronisationsprozesse behandelt. Im Folgenden wird ein vollständiger Informationsprozess, bestehend aus sämtlichen Verarbeitungs-, Zyklusübergangs- und Kommunikationsprozessen, in einem zusammenfassenden Prozessmodell abgebildet. Aus diesem lässt sich eine kausale Ablaufsequenz isolieren, deren minimale und maximale Laufzeiten durch eine Petrinetzsimulation ermittelt werden können. Der kausale Prozess repräsentiert dabei das Verhalten einer ausgewählten Funktionskette in einer verteilten Steuerung. In Abschnitt 5.3.1 wird das beschreibende Systemmodell und in Abschnitt 5.3.2 das daraus ableitbare Simulationsmodell zur vollständigen Laufzeitanalyse vorgestellt.

5.3.1 Systemmodell

Entsprechend der Analyseaufgabe, d. h. der Laufzeitermittlung bei Ausführung einer Funktionskette unter Berücksichtigung sämtlicher Verzögerungsprozesse, wird für das Systemmodell aus Abbildung 5.14 zunächst die funktionale, räumliche und technologische Sicht auf das gesamte *Anwendersystem* herangezogen (Abbildung 5.14, cyan) .

Die Bockmatrizen zur Beschreibung der Struktur des gesamte Anwendersystem, d. h. unter Berücksichtigung aller Partitionierungsprinzipien, stellen eine weitere Aufspaltung der Matrizen $\underline{\underline{C}}_F^+$ und $\underline{\underline{C}}_F^-$ bzw. $\underline{\underline{C}}_{F,S}^+$, $\underline{\underline{C}}_{F,S}^-$ dar (vgl. Abschnitt 5.2.2) und können nach folgendem Schema aufgestellt werden:

$$\underline{\underline{C}}_{F,S,K}^+ = \begin{array}{c} \begin{matrix} \text{Verarbeitende Funktionen} & \text{Zyklus-übergänge} & \text{Kommunikations-funktionen} \end{matrix} \\ \begin{pmatrix} \underline{\underline{C}}_{11} & \underline{\underline{0}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{C}}_{22} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{C}}_{33} \end{pmatrix} \end{array} \begin{matrix} \text{Daten} \\ \text{Nachrichten} \\ \text{Telegramme} \end{matrix}$$

$$= \begin{pmatrix} \underline{\underline{C}}_F^+ & \underline{\underline{0}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \left(\underline{\underline{C}}_P^+ \, \underline{\underline{C}}_P^{+\mathrm{T}}\right) .* \, \underline{\underline{I}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{0}} & \left(\underline{\underline{C}}_G^+ \, \underline{\underline{C}}_G^{+\mathrm{T}}\right) .* \, \underline{\underline{I}} \end{pmatrix} \tag{5.24}$$

mit $\left(\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{x}}^{+}\ \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{x}}^{+\mathrm{T}}\right)=\left(c_{ij}\right)$ mit $c_{ij}\begin{cases}1, & \text{wenn Matrixelement } c_{ij}>0\\ 0, & \text{sonst}\end{cases}$, mit $\boldsymbol{x}\in\{\boldsymbol{P},\boldsymbol{G}\}$

und

$$\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F,S,K}}^{-}=\begin{array}{c}\begin{matrix}\text{Verarbeitende}\\ \text{Funktionen}\end{matrix}\quad\begin{matrix}\text{Zyklus-}\\ \text{übergänge}\end{matrix}\quad\begin{matrix}\text{Kommunikations-}\\ \text{funktionen}\end{matrix}\\ \begin{pmatrix}\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{11}} & \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{12}} & \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{13}}\\ \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{21}} & \underline{\underline{\boldsymbol{0}}} & \underline{\underline{\boldsymbol{0}}}\\ \underline{\underline{\boldsymbol{0}}} & \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{32}} & \underline{\underline{\boldsymbol{0}}}\end{pmatrix}\begin{matrix}\text{Daten}\\ \text{Nachrichten}\\ \text{Telegramme}\end{matrix}\end{array}$$

$$=\begin{pmatrix}\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F}}^{-}-\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{P}}^{-} & \left(\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{PG}}^{-}\ \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{PG}}^{-\mathrm{T}}\right).*\underline{\underline{\boldsymbol{I}}} & \left(\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G}}^{-}\ \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G}}^{-\mathrm{T}}\right).*\underline{\underline{\boldsymbol{I}}}\\ \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{P}}^{-} & \underline{\underline{\boldsymbol{0}}} & \underline{\underline{\boldsymbol{0}}}\\ \underline{\underline{\boldsymbol{0}}} & \left(\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G}}^{-}\ \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G}}^{-\boldsymbol{T}}\right).*\underline{\underline{\boldsymbol{I}}} & \underline{\underline{\boldsymbol{0}}}\end{pmatrix} \tag{5.25}$$

mit $\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{11}}=\left(c_{ij}\right)$ mit $c_{ij}\begin{cases}1, & \text{wenn Matrixelement } c_{ij}>0\\ 0, & \text{sonst}\end{cases}$,

mit $\left(\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{x}}^{-}\ \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{x}}^{-\mathrm{T}}\right)=\left(c_{ij}\right)$ mit $c_{ij}\begin{cases}1, & \text{wenn Matrixelement } c_{ij}>0\\ 0, & \text{sonst}\end{cases}$,

$$\text{mit } \boldsymbol{x}\in\{\boldsymbol{P},\boldsymbol{PG}\},\ \text{mit } \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{PG}}^{-}=\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{P}}^{-}-\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G}}^{-}. \tag{5.26}$$

In den Zeilen stehen der Reihenfolge nach zuerst sämtliche von den verarbeitenden Funktionen erzeugten bzw. empfangenen Daten Dat, dann die zu übermittelnden Nachrichten zwischen den Programmkomponenten Msg und abschließend die von den Gerätekomponenten per Bus gesendeten bzw. empfangenen Telegramme Tel. In den Spalten stehen die verarbeitenden Funktionen Fkt, die Zyklusübergänge $Tran$ sowie die Kommunikationsfunktionen Kom.

Bezogen auf eine spezifisch ausgewählte Funktionskette, deren Laufzeitverhalten bei Ausführung analysiert werden soll, lässt sich eine konkrete *Schaltfolge* von Verarbeitungsfunktionen über den folgenden Spaltenvektor definieren:

$$\underline{\boldsymbol{f}}_{\boldsymbol{seq}}=(f_i)\ \text{mit}\ f_i=\begin{cases}1, & \text{wenn Funktion } i \text{ im Funktionskette vorhanden ist}\\ 0, & \text{sonst}\end{cases}. \tag{5.27}$$

Der entsprechende Zeilenvektor der durch den Funktionsablauf erzeugten Daten kann wie folgt ermittelt werden:

$$\underline{\boldsymbol{d}}_{\boldsymbol{seq}}=\left(d_j\right)=\left(\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F}}^{+}\ \underline{\boldsymbol{f}}_{\boldsymbol{seq}}\right)^{\mathrm{T}}. \tag{5.28}$$

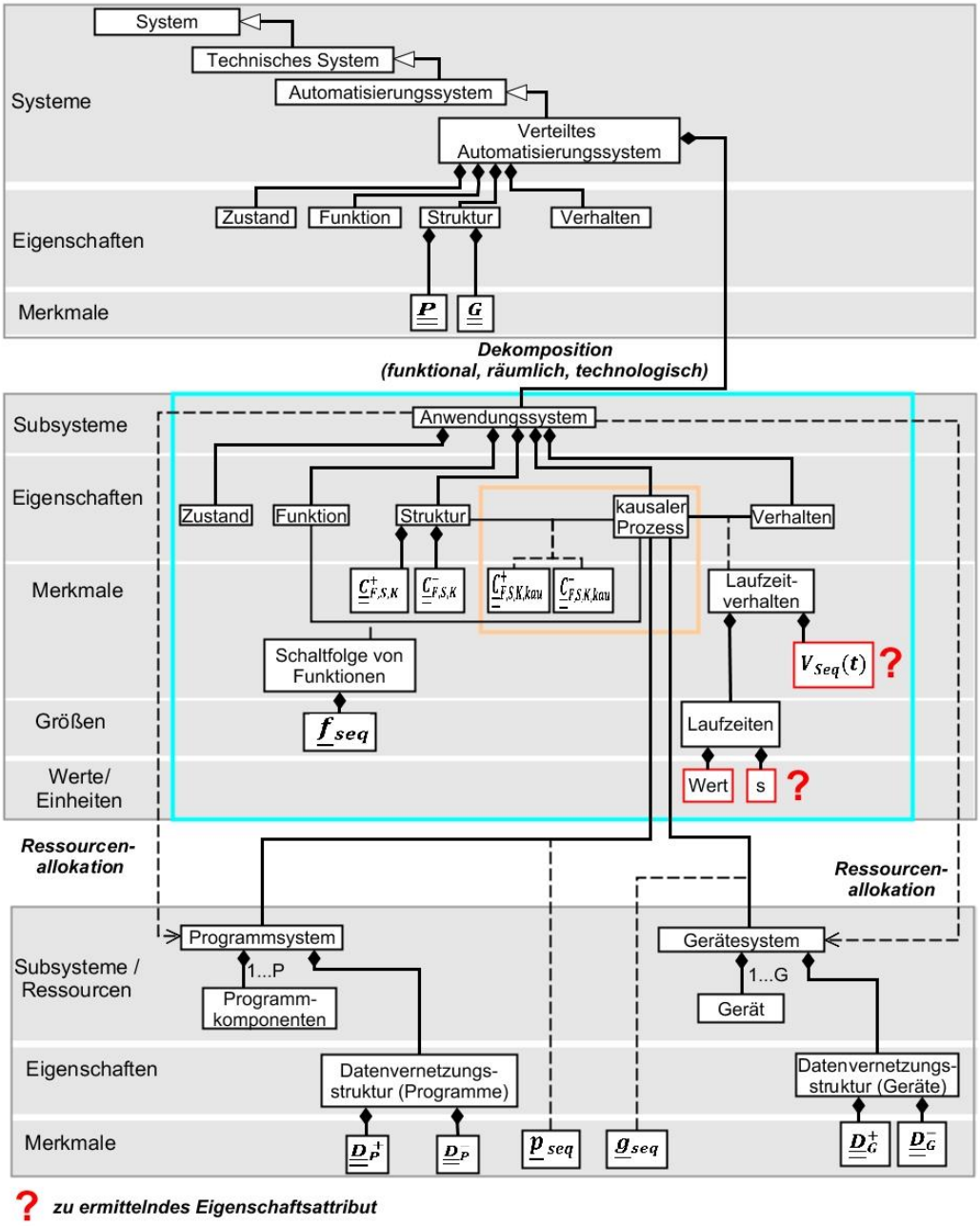


Abbildung 5.14: Systemmodell angepasst an die Problemstellung des Laufzeitverhaltens in einem verteilten Automatisierungssystem mit Anwendungssystem (cyan) und abstrahierten kausalen Ablauf (gelb)

Desweiteren lassen sich die an der Funktionskette beteiligten *Programm- und Gerätekomponenten* ermitteln:

$$\underline{\boldsymbol{p}}_{seq} = \underline{\underline{\boldsymbol{P}}}^{\mathrm{T}}\, \underline{\boldsymbol{f}}_{seq} \tag{5.29}$$

und

$$\underline{\boldsymbol{g}}_{seq} = \underline{\underline{\boldsymbol{G}}}^{\mathrm{T}}\, \underline{\boldsymbol{f}}_{seq}\,. \tag{5.30}$$

Die Vektoren aus (5.29) bzw. (5.30) stellen in Abbildung 5.14 je ein Attribut für eine Assoziationsklasse zwischen dem *kausalen Prozess* und den Ressourcensystemen *Programmsystem* bzw. *Gerätesystem* dar.
Der einer konkreten Schaltfolge $\underline{\boldsymbol{f}}_{\boldsymbol{seq}}$ aus der gesamten Funktionsstruktur $\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F}}^{+}$ und $\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F}}^{-}$ zuzuordnende Prozess $\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F,S,K,kau}}$ kann über ein kausal zusammenhängendes Teilnetz PN strukturell dargestellt werden, für das folgende Eigenschaften gelten:

- Es gibt kein zusammenhängendes Teilnetz PN′ in PN
- Alle Stellen weisen maximal eine Post- und eine Prekante auf:
  $\forall s \in S: |s^{\bullet}| \leq 1 \ \wedge \ |{}^{\bullet}s| \leq 1.$

Unter Berücksichtigung der Zyklusübergänge zwischen den Programmkomponenten und der physikalisch bedingten Kommunikationsprozesse lässt sich der konkrete Prozessablauf als ein kausales Teilnetz darstellen (Abbildung 5.14 und Abbildung 5.16, gelb). Dazu wird zunächst nach demselben Schema gemäß (5.24) und (5.25) vorgegangen wobei lediglich die kausalen Anteile entsprechend der definierten Schaltfolge von Funktionen $\underline{\boldsymbol{f}}_{\boldsymbol{seq}}$ berücksichtigt werden:

$$\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F,S,K,kau}}^{+} = \begin{pmatrix} \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F,kau}}^{+} & \underline{\underline{\boldsymbol{0}}} & \underline{\underline{\boldsymbol{0}}} \\ \underline{\underline{\boldsymbol{0}}} & \left(\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{P,kau}}^{+}\ \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{P,kau}}^{+\,\mathrm{T}}\right) .* \underline{\underline{\boldsymbol{I}}} & \underline{\underline{\boldsymbol{0}}} \\ \underline{\underline{\boldsymbol{0}}} & \underline{\underline{\boldsymbol{0}}} & \left(\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G,kau}}^{+}\ \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G,kau}}^{+\,\mathrm{T}}\right) .* \underline{\underline{\boldsymbol{I}}} \end{pmatrix} \tag{5.31}$$

und

$$\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F,S,K,kau}}^{-} = \begin{pmatrix} \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F,kau}}^{-} - \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{P,kau}}^{-} & \left(\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{PG,kau}}^{-}\ \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{PG,kau}}^{-\,\mathrm{T}}\right) .* \underline{\underline{\boldsymbol{I}}} & \left(\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G,kau}}^{-}\ \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G,kau}}^{-\,\mathrm{T}}\right) .* \underline{\underline{\boldsymbol{I}}} \\ \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{P,kau}}^{-} & \underline{\underline{\boldsymbol{0}}} & \underline{\underline{\boldsymbol{0}}} \\ \underline{\underline{\boldsymbol{0}}} & \left(\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G,kau}}^{-}\ \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G,kau}}^{-\,\mathrm{T}}\right) .* \underline{\underline{\boldsymbol{I}}} & \underline{\underline{\boldsymbol{0}}} \end{pmatrix} \tag{5.32}$$

mit

$$\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{PG,kau}}^{-} = \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{P,kau}}^{-} - \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{G,kau}}^{-}, \tag{5.33}$$

$$\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{x,kau}}^{+} = (c_{ij}) \text{ mit } c_{ij} = \begin{cases} 1, & \text{wenn } \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{x},i,j}^{+}=1 \ \wedge \ \underline{\boldsymbol{f}}_{\boldsymbol{seq},j}=1 \ \wedge \ \underline{\boldsymbol{d}}_{\boldsymbol{seq},i}=1 \\ 0, & \text{sonst} \end{cases}$$

$$\text{mit } (c_i) = \begin{cases} \emptyset, & \text{wenn } \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F,kau},i}^{+}=\underline{\boldsymbol{0}} \\ (c_i), & \text{sonst} \end{cases}$$

$$\text{mit } (c_j) = \begin{cases} \emptyset, & \text{wenn } \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{F,kau},j}^{+}=\underline{\boldsymbol{0}} \\ (c_j), & \text{sonst} \end{cases} \tag{5.34}$$

und

$$\underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{x,kau}} = (c_{ij}) \text{ mit } c_{ij} = \begin{cases} 1, \text{ wenn } \underline{\underline{C}}^{-}_{x,i,j}=1 \ \wedge \underline{\boldsymbol{f}}_{\boldsymbol{seq},j}=1 \ \wedge \underline{\boldsymbol{d}}_{\boldsymbol{seq},i}=1 \\ 0, \text{ sonst} \end{cases}$$

$$\text{mit } (c_i) = \begin{cases} \emptyset, \text{ wenn } \underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{F,kau},i}=\underline{\boldsymbol{0}} \\ (c_i), \text{ sonst} \end{cases}$$

$$\text{mit } (c_j) = \begin{cases} \emptyset, \text{ wenn } \underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{F,kau},j}=\underline{\boldsymbol{0}} \\ (c_j), \text{ sonst} \end{cases} , \qquad (5.35)$$

wobei $\boldsymbol{x} \in \{\boldsymbol{F}, \boldsymbol{P}, \boldsymbol{G}\}$.

Nach der Löschung leerer, irrelevanter Zeilen und Spalten aus (5.31) und (5.32) stehen folgende reduzierte Matrizen zur Verfügung, die die Struktur der Ablaufsequenz innerhalb einer Funktionskette repräsentieren:

$$\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{F,S,K,red}} = \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{F,S,K,kau}} \text{ mit } \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{22},ij} \text{ mit } (c_i) = \begin{cases} \emptyset, \text{ wenn } \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{P,kau},i}=\underline{\boldsymbol{0}} \\ (c_i), \text{ sonst} \end{cases}$$

$$\text{mit } (c_j) = \begin{cases} \emptyset, \text{ wenn } \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{P,kau},j}=\underline{\boldsymbol{0}} \\ (c_j), \text{ sonst} \end{cases}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{33},ij} \text{ mit } (c_i) = \begin{cases} \emptyset, \text{ wenn } \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{G,kau},i}=\underline{\boldsymbol{0}} \\ (c_i), \text{ sonst} \end{cases}$$

$$\text{mit } (c_j) = \begin{cases} \emptyset, \text{ wenn } \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{G,kau},j}=\underline{\boldsymbol{0}} \\ (c_j), \text{ sonst} \end{cases} \qquad (5.36)$$

und

$$\underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{F,S,K,red}} = \underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{F,S,K,kau}} \text{ mit } \underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{12},ij} \text{ mit } (c_j) = \begin{cases} \emptyset, \text{ wenn } (\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{P,kau}}{}^{\mathrm{T}})_j=\underline{0} \\ (c_j), \text{ sonst} \end{cases}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{13},ij} \text{ mit } (c_j) = \begin{cases} \emptyset, \text{ wenn } (\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{G,kau}}{}^{\mathrm{T}})_j=\underline{0} \\ (c_i), \text{ sonst} \end{cases}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{21},ij} \text{ mit } (c_i) = \begin{cases} \emptyset, \text{ wenn } \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{P,kau},i}=\underline{\boldsymbol{0}} \\ (c_i), \text{ sonst} \end{cases}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{32},ij} \text{ mit } (c_i) = \begin{cases} \emptyset, \text{ wenn } \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{G,kau},i}=\underline{\boldsymbol{0}} \\ (c_i), \text{ sonst} \end{cases}$$

$$\text{mit } (c_j) = \begin{cases} \emptyset, \text{ wenn } (\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{P,kau}}{}^{\mathrm{T}})_j=\underline{\boldsymbol{0}} \\ (c_j), \text{ sonst} \end{cases} . \qquad (5.37)$$

5.3.2 Simulationsmodell

Zu Simulationszwecken können die strukturellen Inzidenzmatrizen (5.36) und (5.37) um Parameterangaben ergänzt werden, indem den einzelnen Zellen das entsprechende temporale Verhalten zugeordnet wird. Es erfolgt demnach die Angabe der Bearbeitungszeiten der verarbeitenden Funktionen $\underline{\boldsymbol{t_b}}$ sowie das zuvor ermittelte Verzögerungsverhalten aufgrund der Zyklusübergänge $V_S(t)$ und der Kommunikationsprozesse $V_K(t)$.

Es sei an dieser Stelle angemerkt, dass die in (5.24) hinterlegten verarbeitenden Funktionen je eine Vergröberung aller ihrer Teilfunktionen abbilden. Für die dynamische Betrachtung des Durchlaufens eines konkreten Prozessablaufes sind strenggenommen jedoch nur die Bearbeitungszeiten der am kausalen Prozess beteiligten Teilfunktionen einer verarbeitenden Funktion zu berücksichtigen. Lediglich unter der Annahme einer zyklischen Bearbeitung der verarbeitenden Funktionen, können deren Bearbeitungszeiten mit der ihrer Teilfunktionen gleichgesetzt werden und auch die im kausalen Ablauf verknüpfend wirkenden Teilfunktionen über die Bearbeitungszeiten aus dem Vektor $\underline{\boldsymbol{t_b}}$ beschrieben werden (vgl. Abbildung 5.7).

Anwendungsbeispiel

Bezogen auf das Anwendungsbeispiel aus Abschnitt 1.1.1 soll im Folgenden der Prozessablauf für die Funktionskette aus Abbildung 1.1 bzw. Abbildung 5.15 (je rot dargestellt) in einer verteilten Steuerung simuliert werden, um unter Berücksichtigung sämtlicher Verzögerungen die minimalen und maximalen Laufzeiten dieser Schaltfolge ermitteln zu können. Die hervorgehobene Funktionskette $\underline{\boldsymbol{f_{seq}}}$ aus dem Anwendungsbeispiel repräsentiert die Einleitung eines Heizvorganges mittels Außenluftgerätes aufgrund des Zustandsüberganges in die lange Präsenz und beinhaltet die Präsenzerkennung, die Belegungsauswertung, die Energieauswahl, die Makrofunktion zur Sollwertermittlung und Funktionswahl, die Temperaturregelung, die Stellwertbegrenzung, die Makrofunktion der Außenluftregelung und mehrere Aktorfunktionen. Zunächst kann über die Inzidenzmatrizen des gesamten funktionalen Anwendersystems $\underline{\underline{\boldsymbol{C_F^+}}}$ und $\underline{\underline{\boldsymbol{C_F^-}}}$ und den im Anhang B.1 hinterlegten Programm- und Gerätematrizen das um Zyklusübergänge und Kommunikationsfunktionen erweiterte Petrinetz des Gesamtprozesses (Abbildung 5.16, cyan) erstellt werden. Die Gesamtstruktur aus Abbildung 5.16 wird über die Matrizen $\underline{\underline{\boldsymbol{C_{F,S,K}^+}}}$ und $\underline{\underline{\boldsymbol{C_{F,S,K}^-}}}$ beschrieben.

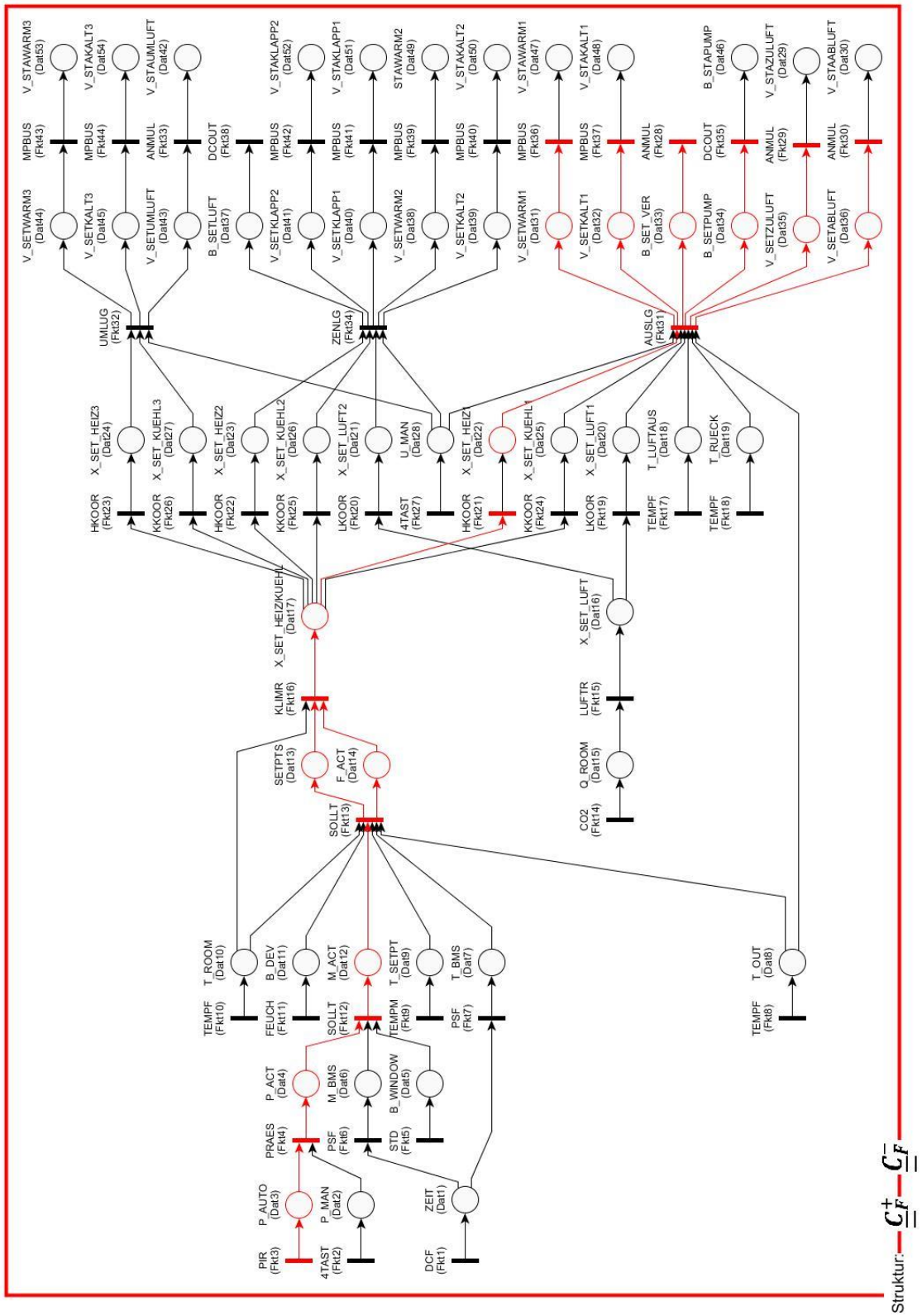


Abbildung 5.15: Funktionverknüpfungen der Raumautomation und exemplarisch hervorgehobene Funktionskette (rot dargestellt)

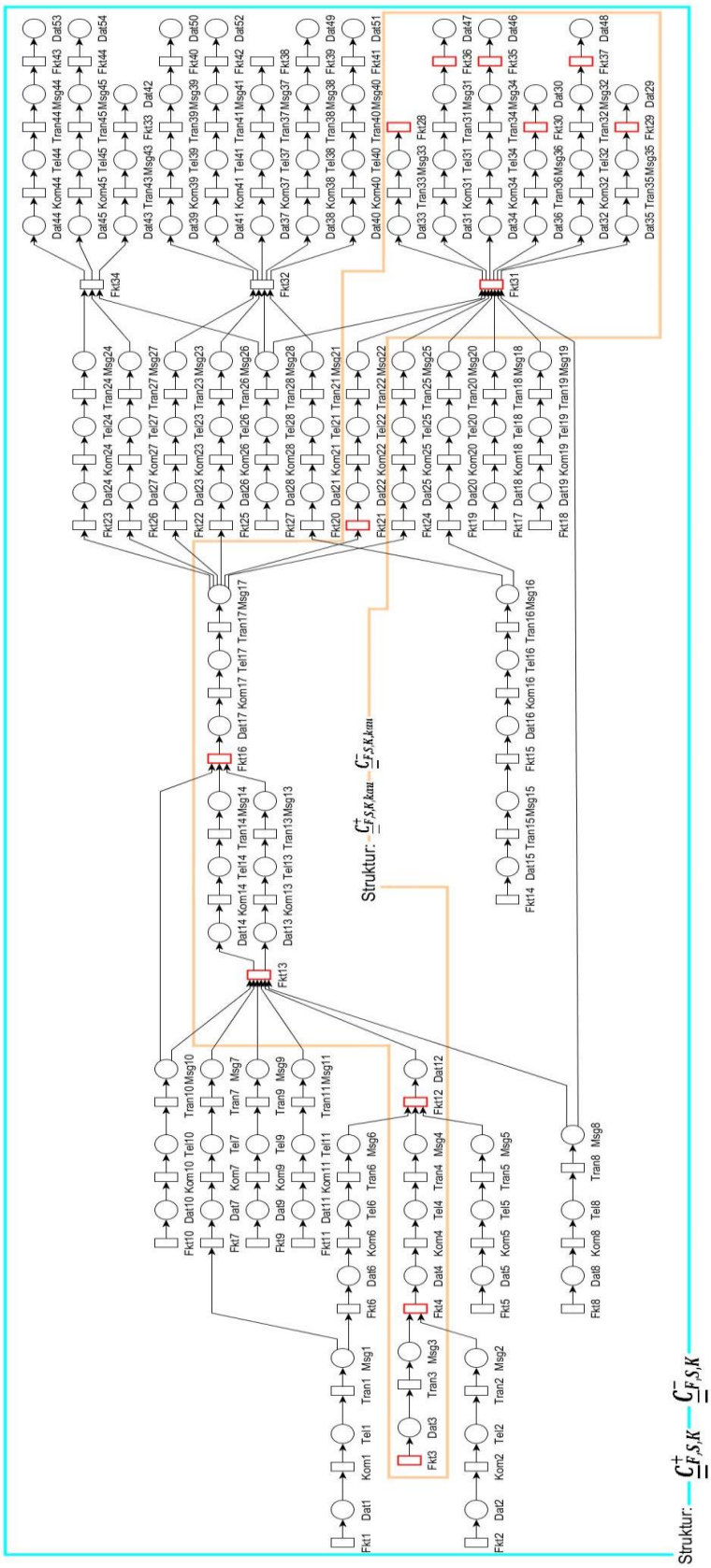


Abbildung 5.16: Petrinetzmodell des Gesamtprozesses (cyan) und eines ausgewählten zu analysierenden Teilprozesses (gelb) aus einer Schaltfolge (rot) unter Berücksichtigung von Zyklusübergängen und Kommunikationsprozessen

Die Petrinetz-Darstellung des entstehenden Prozesses bei Ausführung der Funktionskette

$\underline{\mathbf{f}}_{seq}{}^{\mathrm{T}} = (0\,0\,1\,1\,0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,1\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,1\,1\,1\,0\,0\,0\,1\,1\,1\,0\,0\,0\,0\,0\,0)$

mit den Daten

$\underline{\mathbf{d}}_{seq} = (0\,0\,1\,1\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,0\,0\,1\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,0\,0\,0\,0\,0\,0)$

kann über die Inzidenzmatrizen $\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{F,S,K,kau}}$ und $\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{F,S,K,kau}}$ bzw. $\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{F,S,K,red}}$ und $\underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{F,S,K,red}}$ strukturell beschrieben werden (siehe Anhang B.2). Das entsprechende Petrinetz, als Teilnetz aus dem Gesamtprozess, ist ebenfalls in Abbildung 5.16, gelb bzw. alleinstehend in Abbildung 5.17 gezeigt.

Für eine Simulation des Petrinetzes $\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{F,S,K,kau}}$ und $\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{F,S,K,kau}}$ sind die Transitionen des zeitbehafteten Modells aus Abbildung 5.17 exemplarisch wie folgt parametriert worden. Das Aufrufverhalten der verarbeitenden Funktionen erfolgt gemäß dem kausalen Zusammenhang folgegesteuert:

$$A_{F_4} = A_{F_{12}} = A_{F_{13}} = A_{F_{16}} = A_{F_{21}} = A_{F_{28}} = A_{F_{29}} = A_{F_{30}} = A_{F_{31}} = A_{F_{35}} = A_{F_{36}}$$

$$= A_{F_{37}} = folgegesteuert.$$

Die Bearbeitungszeiten sind konstant definiert als

$$t_{b_3} = t_{b_4} = t_{b_{12}} = t_{b_{13}} = t_{b_{16}} = t_{b_{21}} = t_{b_{28}} = t_{b_{29}} = t_{b_{30}} = t_{b_{31}} = t_{b_{35}}$$

$$= t_{b_{36}} = t_{b_{37}} = 0{,}5\ s.$$

In der Annahme, dass die Zykluszeiten aller beteiligten Programmkomponenten ebenfalls 0,5 s betragen

$$t_{z_3} = t_{z_4} = t_{z_{10}} = t_{z_{13}} = t_{z_{16}} = t_{z_{20}} = t_{z_{21}} = t_{z_{25}} = t_{z_{26}} = t_{z_{27}} = 0{,}5\ \mathrm{s}$$

ergibt sich folgendes Zyklusübergangsverhalten:

$$V_{S\ Fkt3 \rightarrow Fkt4}(t) = V_{S\ Fkt4 \rightarrow Fkt12}(t) = V_{S\ Fkt13 \rightarrow Fkt16}(t) = V_{S\ Fkt16 \rightarrow Fkt21}(t)$$

$$= V_{S\ Fkt21 \rightarrow Fkt31}(t) = V_{S\ Fkt31 \rightarrow Fkt28}(t) = V_{S\ Fkt31 \rightarrow Fkt29}(t) = V_{S\ Fkt31 \rightarrow Fkt30}(t)$$

$$= V_{S\ Fkt31 \rightarrow Fkt35}(t) = V_{S\ Fkt31 \rightarrow Fkt36}(t) = V_{S\ Fkt31 \rightarrow Fkt37}(t) = (uniform \quad 0\ \mathrm{s} \quad 0{,}5\ \mathrm{s}).$$

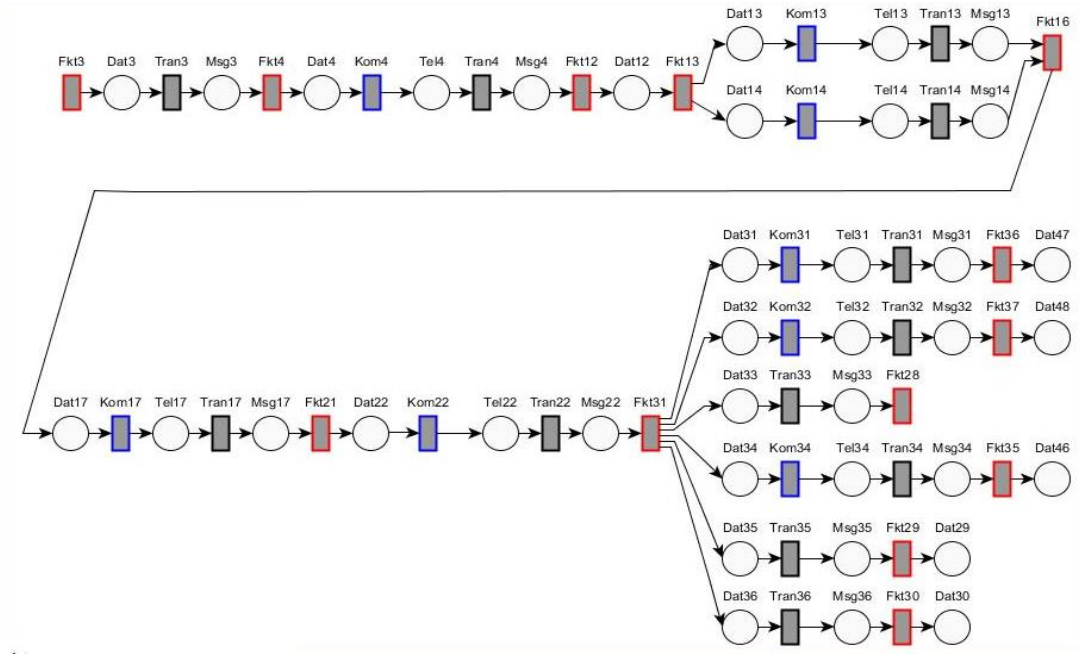


Abbildung 5.17: Exemplarischer kausaler Prozessablauf unter Berücksichtigung von Synchronisations- und Kommunikationsprozessen

Im Falle einer minimalen Busauslastung können die einzelnen Kommunikationsprozesse als deterministisch angesehen werden. Demnach kann die Übertragungszeit eines Telegrammes gemäß Abbildung 5.4, links oben mit 50 ms angegeben werden:

$$V_{K_{4,min}}(t) = V_{K_{13,min}}(t) = V_{K_{14,min}}(t) = V_{K_{17,min}}(t) = V_{K_{22,min}}(t) = V_{K_{31,min}}(t)$$

$$= V_{K_{32,min}}(t) = V_{K_{34,min}}(t) = (deterministisch \quad 50\ \text{ms}).$$

Im Falle einer maximalen Busauslastung ergibt sich demnach für die jeweiligen Kommunikationsprozesse ein lognormales Verhalten (vgl. Abbildung 5.4, rechts unten), sofern angenommen wird, dass allen an den kausalen Prozess beteiligten Telegrammen in ***o*** eine niedrige Priorität zugewiesen ist:

$$V_{K_{4,max}}(t) = V_{K_{13,max}}(t) = V_{K_{14,max}}(t) = V_{K_{17,max}}(t) = V_{K_{22,max}}(t)$$

$$= V_{K_{31,max}}(t) = V_{K_{32,max}}(t) = V_{K_{34,max}}(t) = (lognormal \quad 6{,}5 \quad 0{,}44\ ).$$

In Tabelle 12 und Abbildung 5.18 sind die Simulationsergebnisse nach den einzelnen Prozessschritten bei minimaler und maximaler Busauslastung gezeigt. Für den vollständigen kausalen Prozessablauf ergibt sich demnach eine minimale bzw. maximale Laufzeit von 5,5 s bzw. 15 s.

Tabelle 12: Minimale und maximale ermittelte Laufzeiten des kausalen Prozessablaufes unter Berücksichtigung sämtlicher Verzögerungszeiten bei minimaler und maximaler Busauslastung

	Randbedingungen			
	Minimale Busauslastung		**Maximale Busauslastung**	
Prozess-schritt	Minimale Laufzeit in s	Maximale Laufzeit in s	Minimale Laufzeit in s	Maximale Laufzeit in s
Fkt3	0,5	0,5	0,5	0,5
Tran3	0,5	1	0,5	1
Fkt4	1	1,8	1	1,8
Kom4	1,1	2	1,1	5,6
Tran4	1,2	2,4	1,2	6
Fkt12	1,8	3,2	1,8	6,2
Fkt13	2,2	3,7	2,2	9,7
Fkt16	3	4,6	3	11
Kom17	3,1	4,8	3,1	11,8
Tran17	3,2	5,2	3,2	12,4
Fkt21	3,7	5,8	3,7	12,8
Kom22	3,8	5,9	3,8	13,6
Tran22	3,9	6,3	3,9	13,8
Fkt31	4,3	7	4,5	14,1
Seq	**5,5**	8,1	6,2	**15**

Zur Verdeutlichung der Prozessverläufe aus Abbildung 5.18 lässt sich ein vereinfachter Vergleich mit einem homogenen Poissonprozess aufstellen, der ebenfalls einen diskreten Zählprozess in kontinuierlicher Zeit beschreibt. Demnach lässt ich neben der zeitbezogenen Auftrittswahrscheinlichkeit einer bestimmten Anzahl von Ereignissen (Prozesschritten) auch die Fragestellung der Wahrscheinlichkeit für die Anzahl an Ereignissen innerhalb einer bestimmten Zeit beantworten. Anzumerken ist, dass die Darstellung des homogenen Poissonprozesses gegenüber dem simulierten Verhalten nicht die zeitabhängige Ereignisrate berücksichtigt. Weiterhin variiert die Ereignisrate in Abhängigkeit des aktuell erreichten Prozesschrittes.

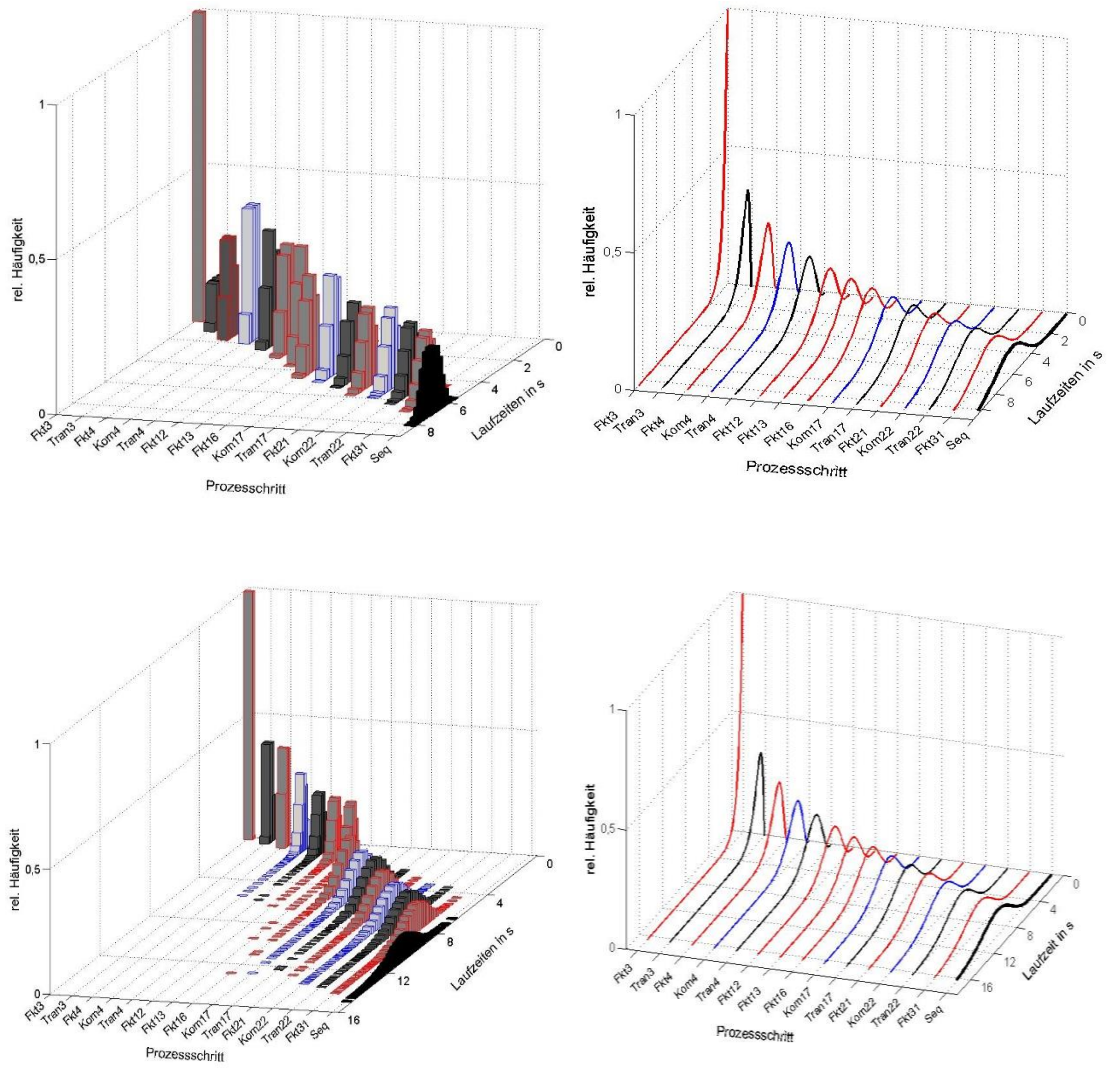


Abbildung 5.18: Simuliertes Laufzeitverhalten (links) einer Schaltfolge von Prozesschritten und vereinfachte Darstellung als homogener Poissonprozess (rechts) bei minimaler (oben) und maximaler (unten) Busauslastung

5.4 Rückführung der Analyseergebnisse in das Systemmodell

In diesem Kapitel wurden für verschiedene Fragestellungen zum dynamischen Verhalten eines verteilten Automatisierungssystems unterstützende beschreibende Systemmodelle sowie simulierbare Petrinetzmodelle aufgezeigt. Zur automatisierten Modellgenerierung wurden die Simulationsmodelle mathematisch beschrieben. Die Modelle wurden validiert und ausgewählte Simulationsergebnisse wurden vorgestellt und diskutiert.

In Abbildung 5.19 sind die Ergebnisse aus den Analysen auszugsweise dargestellt. Das Verzögerungsverhalten eines niederprioren Telegrammes konnte für ein Minimalsystem unter Volllast bestimmt werden. Es weist eine mittlere Verzögerungszeit von 500 ms auf.

Weiterhin konnten aus der Analyse des Zyklusübergangsverhaltens eines Programmsystems Kennwerte für dessen Zeitverhaltens ergründet werden. Konkret ist für einen linearen Verbund mit sechs Programmkomponenten eine mittlere Verzögerungszeit von 4000 ms ermittelt worden.

Für einen exemplarischen Prozessablauf, der sich aus einer gewählten Schaltfolge von Funktionen zusammensetzt, wurden Laufzeiten zwischen 5,5 s und 15 s bestimmt. Das Verhalten des Prozessablauf ist unter Berücksichtigung sämtlicher Verzögerungszeiten simuliert worden und kann vereinfacht durch einen homogenen Poissonprozess beschrieben werden.

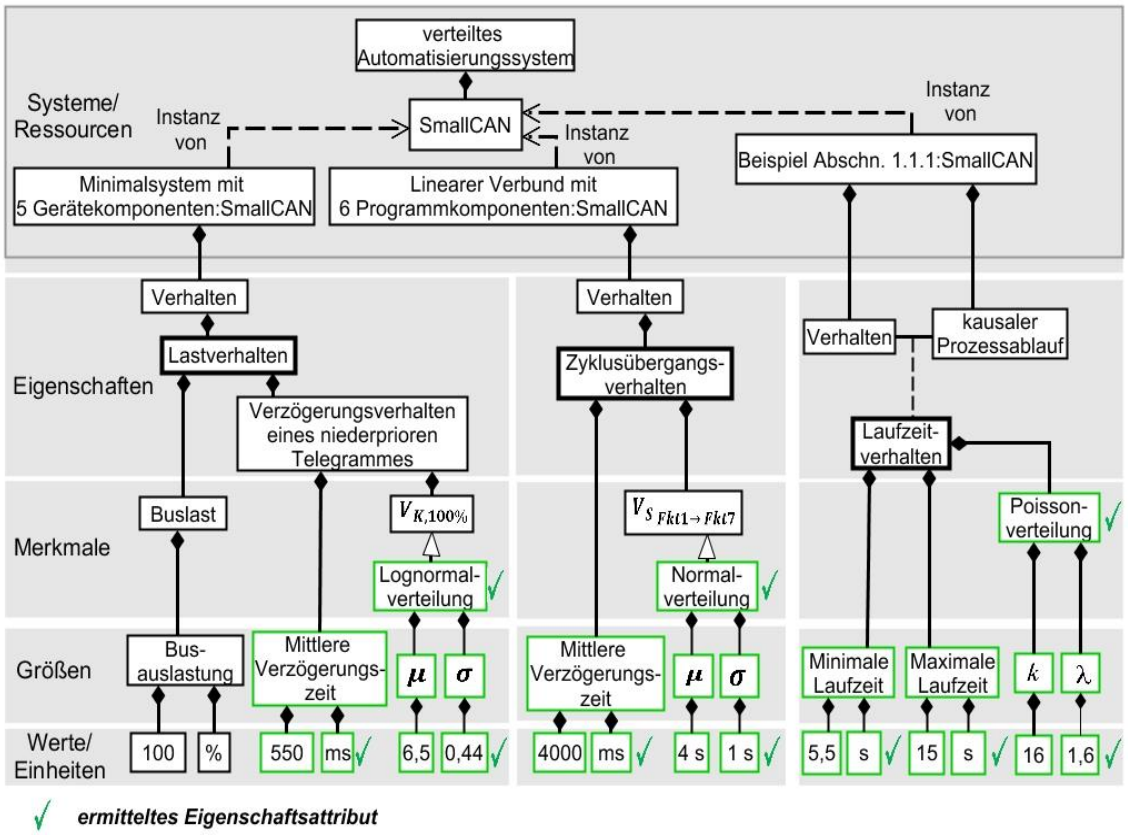


Abbildung 5.19: Ergebnisrückführung in das Systemmodell nach den Analysen zur Zeitbewertung

6 Beobachtbarkeits- und Steuerbarkeitsanalyse

Um Online-Analysen, wie beispielsweise eine Laufzeitüberwachung während des Betriebes betreiben zu können, wird vorausgesetzt, dass das zu analysierende System die wesentliche Eigenschaft der *Beobachtbarkeit* und ggf. auch der *Steuerbarkeit* aufweist. In [Föllinger et al. 1994] werden beide Begriffe als Voraussetzung für die Regelung eines technischen Systems beschrieben. Die Beobachtbarkeit und Steuerbarkeit stellen mit der *Regelbarkeit* demnach das abstrakteste Begriffssystem der Regelungstechnik dar. Nach [Lunze 2012: 150] ist die Steuerbarkeit als die Überführung von jedem Anfangszustand eines Systems in einen gewünschten Zielzustand, mit Hilfe gewählter Eingangsgrößen und innerhalb einer endlichen Zeit definiert. Diese Definition schließt zunächst aus, dass das System auch in diesem Zielzustand gehalten werden kann. Die Beobachtbarkeit aus regelungstechnischer Sicht behandelt die Möglichkeit der Rekonstruktion des Anfangszustandes oder des aktuellen Systemzustandes aus einem bekannten und endlichen Verlauf von Eingangs- und Ausgangsgrößen und dient demnach als Lösung des Beobachtungsproblems. Für lineare zeitinvariante regelungstechnische Systeme, beispielsweise zeitdiskreten Typs ohne Durchgriff gemäß

$$x(k+1) = f\big(x(k), u(k)\big) = \boldsymbol{A}\, x(k) + \boldsymbol{B}\, u(k)$$

$$y(k) = h\big(x(k)\big) = \boldsymbol{C}\, x(k)$$

$$x(0) = x_0\ , \tag{6.1}$$

stellte [Kalman 1959] Kriterien zur Überprüfung der Beobachtbarkeit und Steuerbarkeit auf. Demnach ist ein System genau dann beobachtbar bzw. steuerbar, wenn der Rang folgender Blockmatrizen der Ordnung des Systems entspricht:

$$Rang\left[\boldsymbol{B} \quad \boldsymbol{A}\,\boldsymbol{B} \quad \ldots \quad \boldsymbol{A}^{n-1}\,\boldsymbol{B}\right] = n = \ Ordnung\ des\ Systems \tag{6.2}$$

bzw.

$$Rang\begin{bmatrix} \boldsymbol{C}^{\mathrm{T}} \\ \boldsymbol{C}^{\mathrm{T}}\ A \\ \vdots \\ \boldsymbol{C}^{\mathrm{T}}\ \boldsymbol{A}^{n-1} \end{bmatrix} = n\ =\ Ordnung\ des\ Systems. \tag{6.3}$$

U. a. führten [Cassandras/Lafortune 2008] das Beobachterproblem in die Theorie der Automaten ein. In Analogie zur Zustandsraumdarstellung linearer Systeme nach (6.1) können Automaten demnach wie folgt beschrieben werden:

$$z(k+1) = f\big(z(k), u(k)\big) = \boldsymbol{F}\big(u(k)\big)\, z(k)$$

$$v(k) = h\big(z(k)\big) = \boldsymbol{H}(z(k))$$

$$z(0) = z_0. \tag{6.4}$$

Äquivalent zur Definition der Steuerbarkeit linearer Systeme definiert [Lunze 2012: 429] die Steuerbarkeit eines deterministischen Automaten als die Möglichkeit mit Hilfe geeignet gewählter Eingangsgrößen eine Überführung von jedem Anfangszustand eines Systems in einen gewünschten Zielzustand zu erzwingen.

In der von [Ramadge/Wonham 1989] beschriebenen *Supervisory Control Theory* wird der Begriff der Steuerbarkeit um die Vermeidung des Erreichens verbotener Zustände erweitert. Demnach beinhaltet die Steuerung per *Supervisor* auch ein gezieltes Verhindern bestimmter Zustandsübergänge innerhalb eines Automaten. Die entwickelten Methoden zur Verriegelung von Zustandsübergängen dienten dabei zunächst dem Entwurf von Steuerungen, wobei insbesondere für verteile Systeme eine breitere Anwendung dieser Theorien erst mit der Lösbarkeit der Zustandsraumexplosion, d. h. unter der Verwendung von Petrinetzen als Beschreibungsmittel, vollzogen werden konnte [Giua/Seatzu 2014]. U. a. widmeten sich [Ichikawa/Hiraishi 1988], [Giua 1997], [Schnieder/Chouikha 1998], [Ober 1999], [Frey/Litz 2000] und [Iordache/Antsaklis 2006] dem Steuerungsentwurf mittels Petrinetzen. In [Meda-Campana et al. 2000] werden interpretierte Petrinetze gemäß

$$s(k+1) = f\big(s(k), u(k)\big) = s(k) + \boldsymbol{N}\, t(k) \text{ mit } t(k) \in \mathrm{T}_{akt}(s(k), u(k))$$

$$v(k) = h(s(k))$$

$$s(0) = s_0 \tag{6.5}$$

für die Ermittlung der Beobachtbarkeit und Steuerbarkeit bzw. für die Auslegung eines Beobachters und einer Steuerung herangezogen.

Die in der Literatur vorgestellten Methoden zur Beobachtbarkeit und Steuerbarkeit lassen sich neben der Steuerungssynthese auch für weiterführende Anwendungen wie beispielsweise die Testausführung oder die Überwachung eines Systems nutzen [Isermann 2005] [Tianjing Jiang et al. 2000] [Saki et al. 2011] [Lefebvre 2014].

Analyse der Testbarkeit

In [Jamoussi/Kaminska 1993] und [Ruzicka 2003] wurden die Beobachtbarkeit und Steuerbarkeit als Maß zur Bewertung der Testbarkeit eines Systems herangezogen. Demnach gibt die Testbarkeitsanalyse Auskunft über den Aufwand, der zur Bestimmung eines Zustandes über die Systemausgänge [Bidjan-Irani 1989: 24] bzw. zur Einstellung eines Zustandes über die Systemeingänge nötig ist.

Bezogen auf die Testbarkeit eines Systems stellte [Binder 1994] für die *Beobachtbarkeit* weniger das Beobachtungproblem als vielmehr eine Detektierbarkeit aller Zustände in den Fokus (*Zustandsdetektierbarkeit)*. Demnach kann nach einer getätigten Eingabe am Eingang (Testeingabe) die systeminterne Verarbeitung eindeutig nachvollzogen werden, sofern alle Systemzustände über die Ausgänge des Testobjektes direkt beobachtbar, sprich messbar sind.

Das von [Ramadge/Wonham 1989] erarbeitete Konzept des Supervisors ergänzt das Konzept der *Steuerbarkeit* um die Verriegelung von Zustandsübergängen, wodurch u. a. eine wichtige Voraussetzung für eine kontrollierte Ausführung von Zustandsübergängen innerhalb eines Testprozesses geschaffen wird. Sofern das zu testendes System über seine Eingänge steuerbar ist, d. h. auch unerwünschte bzw. selbstinitiierende Zustandsübergänge verhindert werden können (*Zustandssteuer-barkeit*), lässt sich eine Beurteilung nach einer Testeingabe auf Basis der eindeutig zuzuordnenden Ausgänge durchführen.

Analyse der Laufzeitüberwachbarkeit

Für die Laufzeitüberwachung eines Steuerungsprozesses während der Betriebsphase kann ein Monitor angewendet werden, der nach dem Beobachtbarkeitskriterium der *Ereignisdetektierbarkeit* arbeitet. Für den Fall dass einzelne Ereignisse innerhalb des Steuerungsprozesses nicht detektierbar sind, stellt [Aguirre-Salas/Santoyo-Sanchez 2009] Kriterien für die *Sequenzdetektierbarkeit*, d. h. für eine indirekte Beobachtung dieser Ereignisse, auf.

Für die *Steuerbarkeit* eines sich im laufenden Betrieb befindlichen Systems gilt: das mittels Monitor zu beobachtende System soll möglichst geringfügig beeinflusst werden [Kurschl 2000: 72], d. h. ein Eingriff über ein Analysewerkzeug ist im Falle der Laufzeitüberwachung zu vermeiden.

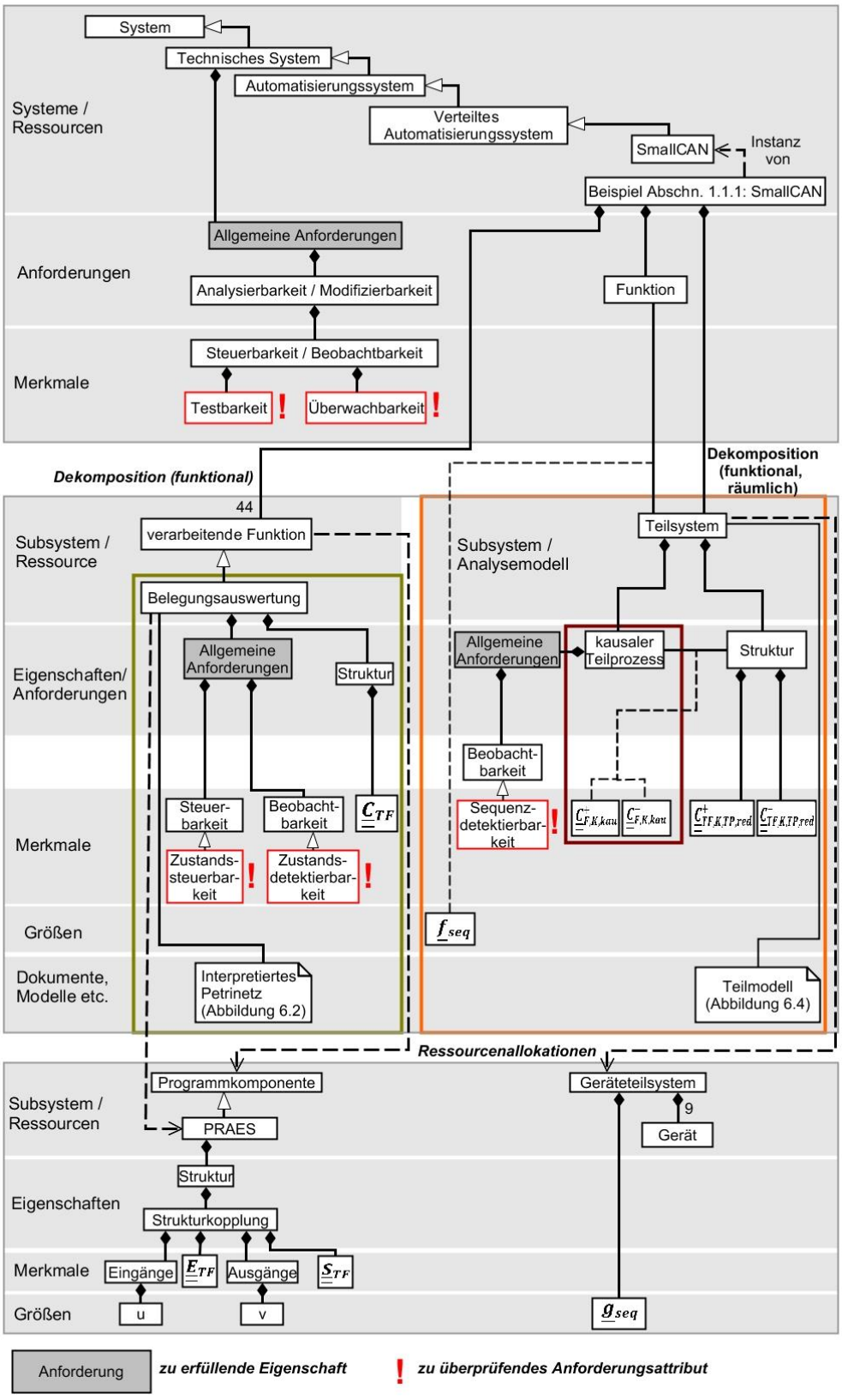


Abbildung 6.1: Systemmodell zur Darstellung der in diesem Kapitel behandelten Beobachtbarkeits-und Steuerbarkeitsanalysen an den Beispielen der funktional betrachteten Funktion zur Belegungsauswertung (grün) und eines funktional räumlich betrachteten Teilprozesses (orange) bzw. eines abstrahierten kausalen Prozesses (dunkelrot)

In den folgenden Abschnitten werden die durchgeführten Beobachtbarkeits- und Steuerbarkeitsanalysen zur Bewertung der Testbarkeit eines Systems sowie zur Bewertung der Realisierbarkeit einer Laufzeitüberwachung vorgestellt. Das den Analysen zugrundeliegende Systemmodell kann Abbildung 6.1 entnommen werden. Zur Bewertung der Testbarkeit einer verarbeitenden Funktion werden in Abschnitt 6.1 und Abschnitt 6.2 die Zustandsdetektierbarkeit und die Zustandssteuerbarkeit am Beispiel der in grün umrandeten Funktion zur Belegungsauswertung behandelt. In Abschnitt 6.3 wird auf Basis eines funktional und räumlich betrachteten Teilsystems bzw. Analysemodells (orange) die Sequenzdetektierbarkeit eines kausalen Ablaufes (dunkelrot) diskutiert.

6.1 Zustandsdetektierbarkeit am Beispiel einer verarbeitenden Funktion

Mit der Darstellung der Verarbeitungsfunktion der Belegungsauswertung aus Abbildung 1.2 in Form einer Automatengleichung gemäß (6.4) lässt sich die Zustandsdetektierbarkeit am einfachsten verifizieren, da dieser alle möglichen Systemzustände in ihrer klarsten Form abbildet. Sofern diesen Systemzuständen unterscheidbare Ausgänge zugeordnet werden können, gilt das System, in diesem Fall die Funktion der Belegungsauswertung, als vollständig beobachtbar.
Für die Prüfung der Zustandsdetektierbarkeit von Petrinetzen können u. a. folgende Ansätze verfolgt werden:

- Ein Petrinetz, in dem jeder Transition höchstens eine Vor- bzw. Nachstelle zugewiesen wird und in dem jede Stelle mit höchstens einer Marke belegt ist, kann durch Weglassen der Transitionen in seinen Zustandsgraphen überführt werden [Lunze 2012: 401] der alle möglichen Systemzustände abbildet.

- Ein Petrinetz in dem jeder Transition genau eine Vor- bzw. Nachstelle zugewiesen wird (Abbildung 2.3, links oben) wird als Zustandsmaschine (Automat) bezeichnet [Baumgarten 1997] und ist gemäß (6.4) beschreibbar und hinsichtlich der Beobachtbarkeit analysierbar.

- Erfüllt das Petrinetz nicht die Kriterien einer Zustandsmaschine weist aber einen determinierten Erreichbarkeitsgraphen auf (vgl. Abbildung 2.3, recht oben), sind durch dessen Markierungen alle möglichen Systemzustände abgebildet. Der endliche Erreichbarkeitsgraph repräsentiert somit unmittelbar einen Automatengraphen [Litz 1995].

- Liegt eine Darstellung des Steuerungsprozesses in Form eines interpretierten Petrinetzes gemäß (6.5) vor, kann auf die Erstellung des Erreichbarkeitsgraphen verzichtet werden und zur Analyse der Zustandsdetektierbarkeit das in [Ramirez-Trevino et al. 2003] vorgestellte Theorem der Markendetektierbarkeit herangezogen werden. Demnach ist ein interpretiertes Petrinetz markendetektierbar, wenn alle Stellen im Petrinetz messbar sind und das Petrinetz selbst ereignisdetektierbar ist.

Für das interpretierte Petrinetz aus Abbildung 6.2 kann die Zuordnung der Stellen zu den Ausgängen über die folgende Ausgangsmatrix beschrieben werden:

$$\underline{\underline{\boldsymbol{S_{TF}}}} = \begin{matrix}\text{Stellen}\\ \begin{pmatrix} 1&0&0&0&0\\ 0&1&0&0&0\\ 0&0&1&0&0\\ 0&0&0&1&0\\ 0&0&0&0&1 \end{pmatrix}\end{matrix} \begin{matrix}\\ \text{Ausgänge (v1 bis v5)}\end{matrix} . \tag{6.6}$$

Es wird zunächst ersichtlich, dass sämtliche Stellen über die Ausgänge v(P1)=v1 bis v(P5)=v5 messbar sind. In diesem Falle ergibt sich die äquivalente Matrix (6.7), die die messbaren Stellen gegenüber allen Stellen bzw. internen Daten des Petrinetzes abbildet.

$$\underline{\underline{\boldsymbol{\varphi}}} = \begin{matrix}\text{Stellen}\\ \begin{pmatrix} 1&0&0&0&0\\ 0&1&0&0&0\\ 0&0&1&0&0\\ 0&0&0&1&0\\ 0&0&0&0&1 \end{pmatrix}\end{matrix} \begin{matrix}\\ \text{messbare}\\ \text{Stellen}\end{matrix} \tag{6.7}$$

Die innere Struktur der zu analysierenden verarbeitenden Funktion wird über ihre Inzidenzmatrix aus (6.8) beschrieben.

$$\underline{\underline{\boldsymbol{C_{TF}}}} = \begin{matrix}\text{Teilfunktionen}\\ \begin{pmatrix} -1&0&0&0&0&1&0&0&0&0&0&1&0&1\\ 1&-1&0&0&-1&-1&-1&-1&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&1&0&0&-1&-1&-1&0\\ 0&0&1&-1&-1&-1&0&-1&1&-1&-1&-1&0&0\\ 0&0&0&0&1&0&1&0&0&0&1&0&1&-1 \end{pmatrix}\end{matrix} \begin{matrix}\\ \text{Stellen}\end{matrix} \tag{6.8}$$

Gemäß [Rivera-Rangel et al. 2005] wird die Ereignisdetektierbarkeit auf Grundlage des Produktes $\underline{\underline{\boldsymbol{\varphi}}}\,\underline{\underline{\boldsymbol{C_{TF}}}}$ der Matrizen (6.7) und (6.8) analysierbar.

$$\underline{\underline{\boldsymbol{\varphi}}}\,\underline{\underline{\boldsymbol{C_{TF}}}} = \begin{matrix}\text{Teilfunktionen}\\ \begin{pmatrix} -1&0&0&0&0&1&0&0&0&0&0&1&0&1\\ 1&-1&0&0&-1&-1&-1&-1&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&1&0&0&-1&-1&-1&0\\ 0&0&1&-1&-1&-1&0&-1&1&-1&-1&-1&0&0\\ 0&0&0&0&1&0&1&0&0&0&1&0&1&-1 \end{pmatrix}\end{matrix} \begin{matrix}\\ \text{messbare}\\ \text{Stellen}\end{matrix} \tag{6.9}$$

Die Ergebnismatrix stellt die ereignisgeknüpften Teilfunktionen in Relation zu den messbaren Stellen. Folgende Bedingungen für die Ereignisdetektierbarkeit sind an diese Matrix zu stellen:

- Alle Spalten der Matrix $\underline{\underline{\boldsymbol{\varphi}}}\,\underline{\underline{\boldsymbol{C}_{TF}}}$ sind nicht leer.
- Für gleiche Spalten der Matrix $\underline{\underline{\boldsymbol{\varphi}}}\,\underline{\underline{\boldsymbol{C}_{TF}}}$ gilt: Die den Spalten zugeordneten Transitionen repräsentieren keine internen Zustandsübergänge und die Transitionen weisen unterscheidbare Eingangsinformationen auf.

Die Matrix (6.9) enthält keine leeren Spalten und lediglich die Spalten 3 und 9 bzw. 4 und 10 sind mit gleichen Inhalten gefüllt. Demnach müssen die den gleichgefüllten Spalten zugehörigen Transitionen T3 und T9 bzw. T4 und T10 unterschiedliche Eingangsinformationen, die sich aus den Eingängen selbst und den jeweiligen Vorbereichen der Transitionen zusammensetzen, zugewiesen sein. Die Zuordnung der internen Transitionen gegenüber den Eingangsinformationen erfolgt über die Eingangsmatrix:

$$\underline{\underline{\boldsymbol{E}_{TF}}} = \overset{\text{Teilfunktionen}}{\begin{pmatrix} 1&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&1&0 \end{pmatrix}} \text{Eingänge (u1 bis u11)} \quad . \tag{6.10}$$

Zwar sind die Eingänge u(T3)=u3=u(T9)=u8: ¬P_AUTO-0 bzw. u(T4)=u4=u(T10)=u9: ¬P_MAN unter Nichtberücksichtigung interner Verarbeitungs- bzw. Wartezeiten gleich (Abbildung 6.2, rechts oben), die Vorbereiche ·T3=P2≠·T9=P3 bzw. ·T4=P2≠·T10=P3 sind jedoch verschieden, wodurch insgesamt betrachtet die Marken- bzw. Zustandsdetektierbarkeit des Steuerungsprozesses zur Belegungsauswertung erfüllt ist.

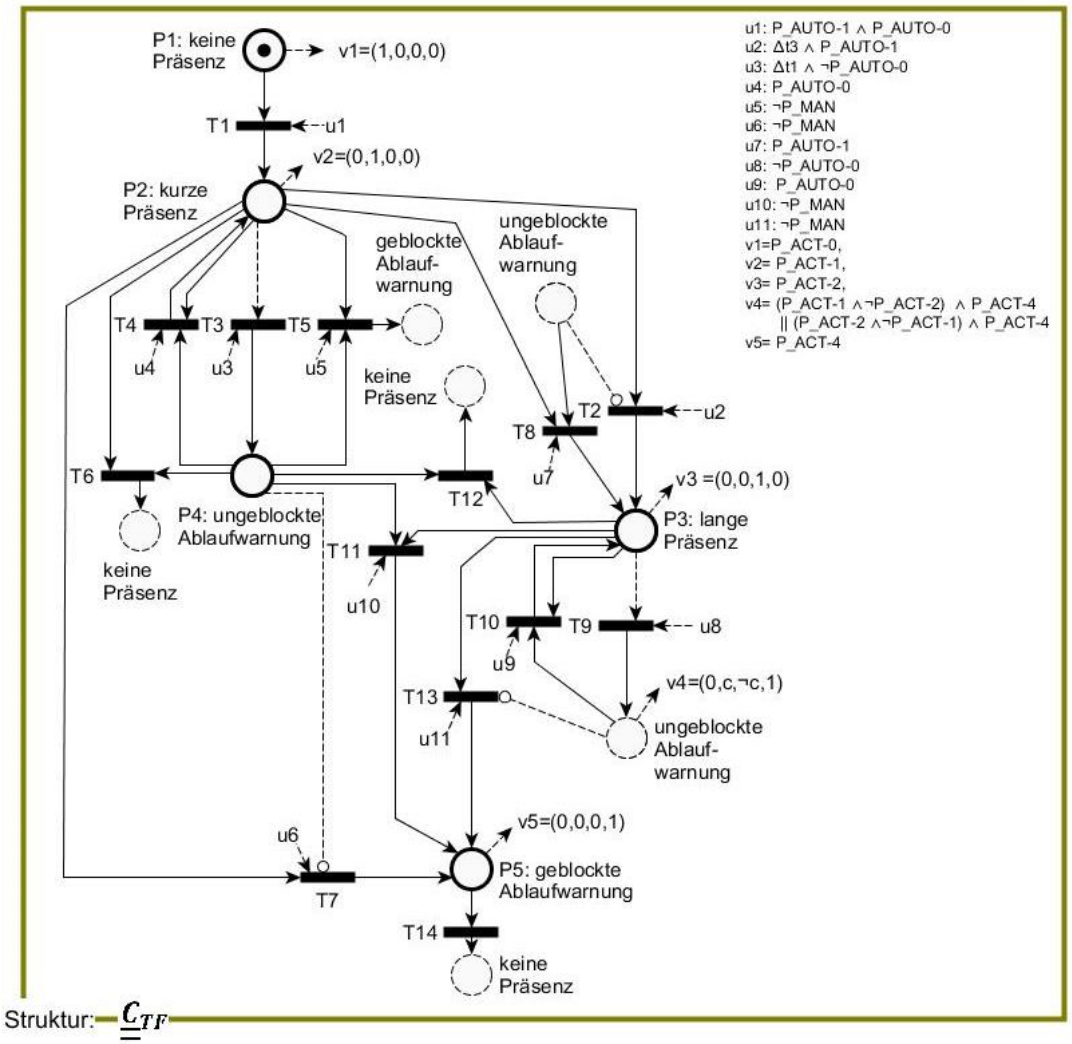


Abbildung 6.2: Interpretiertes Petrinetz eines exemplarischen Steuerungsprozesses

6.2 Zustandssteuerbarkeit am Beispiel einer verarbeitenden Funktion

Für die Steuerbarkeit der Funktion zur Belegungsauswertung kann in Anlehnung an den in [Ichikawa/Hiraishi 1988] vorgestellten Algorithmus eine Kontrollstruktur eingeführt werden. Diese basiert auf der Erweiterung des Steuerungsprozesses um externe Eingangsstellen, die mit der Menge derjenigen Transitionen gekoppelt werden, die aus den (Zwischen-)Zielmarkierungen gewünschten Ablauffolgen heraus aktivierbar sind. Entsprechend der Auftrittshäufigkeit von Transitionen innerhalb der gewünschten Ablauffolge, die beispielsweise einen definierten Testfall darstellt, werden die externe Eingangsstellen mit keiner, einer oder mehreren Marken belegt. Sofern also innerhalb der Abfolge eine an die Kontrollstruktur geknüpfte Transition nicht vorkommt, bleibt zur Vermeidung ihres selbsttätigen Schaltens die entsprechende externe Eingangsstelle unbelegt.

In Abbildung 6.3 ist der um eine Kontrollstruktur zu erweiternden Steuerungsprozess aus Abbildung 6.2 dargestellt, um gezielte Zustandsübergänge ab der Anfangsmarkierung (M0: keine Präsenz) über die Zwischenmarkierungen (M1 bzw. M3: kurze Präsenz) bis zu den Zielmarkierungen (M2 bzw. M4: lange Präsenz) herbeizuführen.

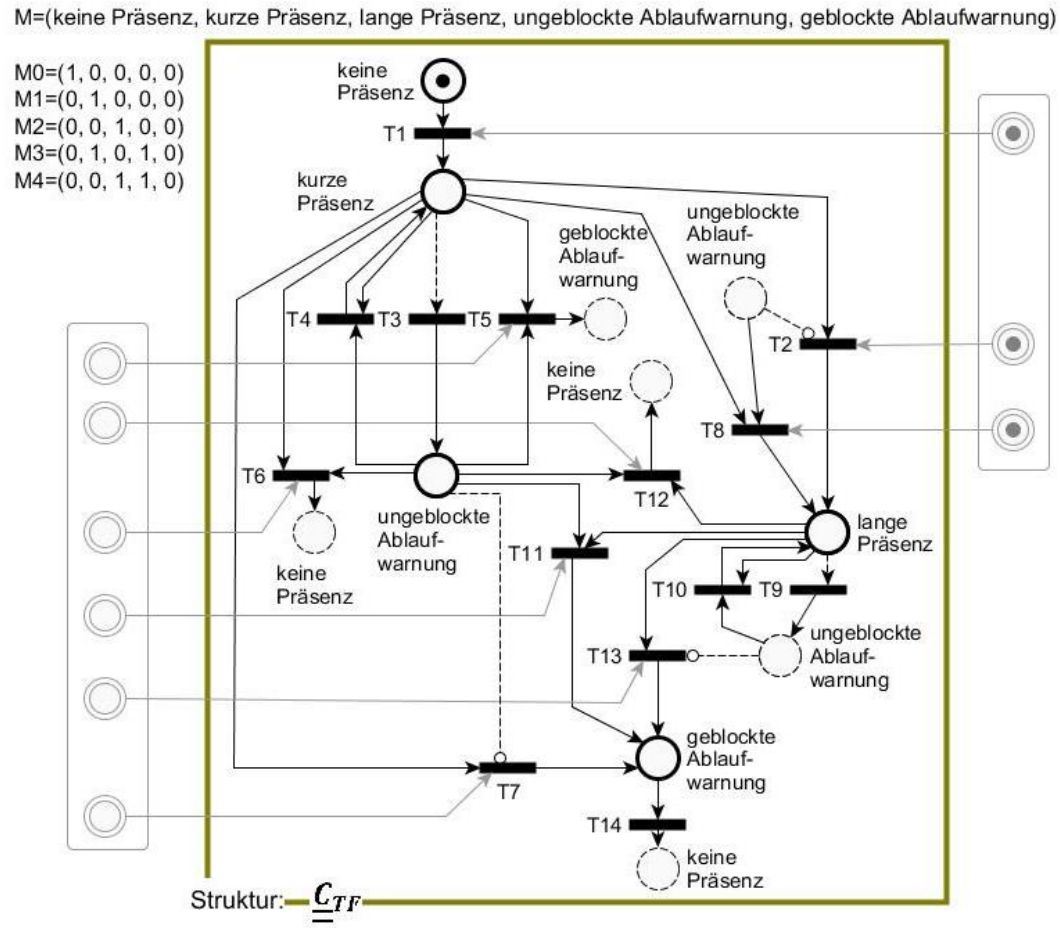


Abbildung 6.3: Exemplarischer Steuerungsprozess mit Kontrollstruktur

Durch Schalten von T1 wird zunächst ein Zustandsübergang in die kurze Präsenz (M1) eingeleitet. Durch die ergänzte Kontrollstruktur wird das Schalten der Transitionen T5, T6, T7 unterbunden, wodurch eine Aufrechterhaltung der kurzen Präsenz innerhalb der Markierungen M1 bzw. M3 erreicht wird. Aus diesen Markierungen heraus kann durch das Schalten von T2 bzw. T8 ein Zustandswechsel in die lange Präsenz (M2) ermöglicht werden, da die Kontrollstruktur je ein einmaliges Schalten dieser Transitionen erlaubt. Die resultierenden Zielmarkierungen M2 bzw. M4 werden wiederum aufrechterhalten, indem das Schalten der Transitionen T11, T12 und T13 durch die Kontrollstruktur verhindert wird.

6.3 Sequenzdetektierbarkeit am Beispiel eines kausalen Prozessablaufes

Für die Beobachtbarkeit eines kausalen Prozessablaufes (Prozesszustandserfassung), dessen Prozessschritte nur zum Teil detektierbar sind, stellt [Aguirre-Salas/Santoyo-Sanchez 2009] ein Theorem auf, welches auf der Detektierbarkeit von Sequenzen beruht. Demnach bilden benachbarte oder einzelne, nach außen nicht unterscheidbare und nicht sichtbare Transitionen (nicht detektierbare Prozesschritte) eine Gruppe, für die folgende Bedingungen für deren Detektierbarkeit erfüllt sein müssen:

- Es existiert genau ein Pfad zwischen jedem Paar von Eingangs- und Ausgangstransitionen der Gruppe.
- In der Anfangsmarkierung des zu analysierenden Petrinetzes hat keine Transition innerhalb der Gruppe Konzession.
- Sobald eine Eingangstransition t_r der Gruppe schaltet, darf keine weitere Eingangstransition t_x der Gruppe schalten, solange sich die von t_r erzeugten Marken noch innerhalb der Gruppe befinden.

Wird vorausgesetzt, dass ein Monitor an das Kommunikationssystem des Steuerungssystems angeschlossen wird, können zunächst sämtliche Zustände detektiert werden, die das Absenden eines Bustelegrammes nach sich ziehen. Daher eignet sich für die Abbildung des kausalen Prozessablaufes, der beobachtet werden soll, folgende Strukturdarstellung:

$$\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{F,K,kau}} = \begin{pmatrix} \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{11}} & \underline{\underline{\boldsymbol{0}}} \\ \underline{\underline{\boldsymbol{0}}} & \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{22}} \end{pmatrix} \begin{matrix} \text{verarbeitende Funktionen} \\ \text{Daten} \end{matrix} \quad \begin{matrix} \text{Kommunikations-funktion} \\ \text{Telegramme} \end{matrix}$$

$$= \begin{pmatrix} \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{F,kau}} & \underline{\underline{\boldsymbol{0}}} \\ \underline{\underline{\boldsymbol{0}}} & \left(\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{G,kau}}\, \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{G,kau}}{}^{\mathrm{T}}\right) .* \underline{\underline{\boldsymbol{I}}} \end{pmatrix} \tag{6.11}$$

und

$$\underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{F,K,kau}} = \begin{pmatrix} \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{11}} & \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{12}} \\ \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{21}} & \underline{\underline{\boldsymbol{0}}} \end{pmatrix} \begin{matrix} \text{verarbeitende Funktionen} \\ \text{Daten} \end{matrix} \quad \begin{matrix} \text{Kommunikations-funktion} \\ \text{Telegramme} \end{matrix}$$

$$= \begin{pmatrix} \underline{\underline{C}}^{-}_{F,kau} - \underline{\underline{C}}^{-}_{G,kau} & \left(\underline{\underline{C}}^{-}_{G,kau}\, \underline{\underline{C}}^{-}_{G,kau}{}^{\mathrm{T}}\right) .* \underline{\underline{I}} \\ \underline{\underline{C}}^{-}_{G,kau} & \underline{\underline{0}} \end{pmatrix}. \tag{6.12}$$

Die Strukturbeschreibung aus (6.11) und (6.12) lässt sich in Analogie zu dem in Abschnitt 5.2.2 eingeführten Schema entwickeln, berücksichtigt aber anstatt der Zyklusübergänge nun die von Monitor auswertbaren Kommunikationsfunktionen.

In Abbildung 6.4 ist der kausale Ablauf gemäß (6.11) und (6.12) für das in Abschnitt 5.3 eingeführte Beispiel in Schwarz skizziert. Für die Bewertung der Sequenzdetektierbarkeit dieses kausalen Ablaufes ist jedoch die in Abbildung 6.4 skizzierte Gesamtstruktur (schwarze und graue Petrinetzelemente) abzubilden, die ergänzend die übrigen einflussnehmenden Elemente beinhaltet, so dass auf Basis der über Kommunikationsfunktionen detektierbaren Zustände auch auf die vorangegangenen Zustandswechsel geschlossen werden kann. Die Gesamtstruktur lässt sich über folgendes Konstrukt ermitteln:

$$\underline{\underline{C}}^{+}_{TF,K,TP} = \begin{array}{c} \begin{matrix} \text{verarbeitende Teilfunktionen} & \text{Kommunikations-funktionen} & \text{verarbeitende Teilfunktionen} & \text{Kommunikations-funktionen} \end{matrix} \\ \begin{pmatrix} \underline{\underline{C}}_{11} & \underline{\underline{0}} & \underline{\underline{C}}_{13} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{C}}_{22} & \underline{\underline{0}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{C}}_{34} \end{pmatrix} \begin{matrix} \text{Daten} \\ \text{Telegramme} \\ \text{Telegramme} \end{matrix} \end{array}$$

$$= \begin{pmatrix} \underline{\underline{C}}^{+}_{F,kau} & \underline{\underline{0}} & \underline{\underline{C}}^{+}_{TF} & \underline{\underline{0}} \\ \underline{\underline{0}} & \left(\underline{\underline{C}}^{+}_{G,kau}\, \underline{\underline{C}}^{+}_{G,kau}{}^{\mathrm{T}}\right) .* \underline{\underline{I}} & \underline{\underline{0}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{C}}^{+}_{K,Seq} \end{pmatrix} \tag{6.13}$$

und

$$\underline{\underline{C}}^{-}_{TF,K,TP} = \begin{array}{c} \begin{matrix} \text{verarbeitende Teilfunktionen} & \text{Kommunikations-funktionen} & \text{verarbeitende Teilfunktionen} & \text{Kommunikations-funktionen} \end{matrix} \\ \begin{pmatrix} \underline{\underline{C}}_{11} & \underline{\underline{C}}_{12} & \underline{\underline{0}} & \underline{\underline{0}} \\ \underline{\underline{C}}_{21} & \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{C}}_{33} & \underline{\underline{0}} \end{pmatrix} \begin{matrix} \text{Daten} \\ \text{Telegramme} \\ \text{Telegramme} \end{matrix} \end{array}$$

$$= \begin{pmatrix} \underline{\underline{C}}^{-}_{F,kau} - \underline{\underline{C}}^{-}_{G,kau} & \left(\underline{\underline{C}}^{-}_{G,kau}\, \underline{\underline{C}}^{-}_{G,kau}{}^{T}\right) .* \underline{\underline{I}} & \underline{\underline{0}} & \underline{\underline{0}} \\ \underline{\underline{C}}^{-}_{G,kau} & \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{0}} & \left(\underline{\underline{C}}^{+}_{TF}{}^{\mathrm{T}}\, \underline{\underline{C}}^{+}_{TF}\right) .* \underline{\underline{I}} & \underline{\underline{0}} \end{pmatrix} \tag{6.14}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{TF}} \text{ mit } (c_j) = \begin{cases} \underline{\boldsymbol{C}}^{+}_{\boldsymbol{F,kau,j}}, & \text{wenn } \sum_{i=1}^{D} \underline{\boldsymbol{C}}^{-}_{F,ij} > \sum_{i=1}^{D} \underline{\boldsymbol{C}}^{-}_{\boldsymbol{F,kau},ij} \\ \underline{\mathbf{0}}, & \text{sonst} \end{cases} \tag{6.15}$$

und

$$\text{mit } \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{K,Seq}} = (\underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{G,Seq}} - \underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{G,kau}})^{T} \tag{6.16}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{G,Seq}} \text{ mit } (c_j) = \begin{cases} \underline{\boldsymbol{C}}^{-}_{G,j}, & \text{wenn } (\underline{\mathbf{f}}_{\boldsymbol{seq}}{}^{\mathrm{T}})_j > 0 \\ \underline{\mathbf{0}}, & \text{sonst} \end{cases}. \tag{6.17}$$

Die Struktur des bereits eingeführten kausalen Ablaufes nach (6.11) und (6.12) findet sich in den ersten beiden Spalten der Blockmatrizen (6.13) und (6.14) wieder, wobei die erste Spalte strenggenommen lediglich die dem kausalen Ablauf zugehörigen direkten Pfade (Teilfunktionen) aus den verarbeitenden Funktionen repräsentiert. D. h. bezogen auf das Beispiel aus Abbildung 6.4 bilden die Teilfunktionen Fkt4.1 bis Fkt31.1 nur diejenigen internen Transitionen der Funktionen Fkt4 bis Fkt31 ab, deren Vorbereich und Nachbereich Bestandteil des kausalen Ablaufes sind. Die vergröberten, in Abbildung 6.4 grau dargestellten Transitionen Fkt4.2 bis Fkt31.6 fassen diejenigen Teilfunktionen aus Fkt4 bis Fkt31 zusammen, für die folgende Charakteristika gelten:

- Der Nachbereich der Teilfunktion ist Bestandteil des kausalen Prozessablaufes.
- Der Vorbereich der Teilfunktion ist nicht Bestandteil des kausalen Prozessablaufes.
- Der Vorbereich der Teilfunktion ist über eine Kommunikationsfunktion markierbar.

Die in grau dargestellten Teilfunktionen Fkt4.2 bis Fkt31.6 sind demnach nicht direkter Bestandteil des kausalen Prozesses, können aber durch den verknüpften Nachbereich einen Einfluss auf das Verhalten des Funktionsablaufes haben. Dieser Umstand ist bei der Analyse der Sequenzdetektierbarkeit zu berücksichtigen. Die Relationen der Teilfunktionen Fkt4.2 bis Fkt31.6 zu ihren Nachbereichen, d. h. den Daten des kausalen Prozesses, werden in der dritten Spalte aus (6.13) beschrieben. In der dritten Spalte aus (6.14) sind die Verknüpfungen der Teilfunktionen Fkt4.2 bis Fkt31.6 zu den grau hinterlegten Vorbereichen (Tel2 bis Tel28) abgebildet. Die Markierung der Vorbereiche erfolgt über die in der vierten Spalte hinterlegten Kommunikationsfunktionen Kom2 bis Kom28. Diese sind als nicht direkter Bestandteil des kausalen Prozessablaufes in Abbildung 6.4 ebenfalls grau dargestellt. Die Zuordnung der Kommunikationsfunktionen Kom2 bis Kom28 zu den vergröberten Stellen Tel2 bis Tel25 erfolgt gemäß (6.16).

Die dritte Zeile der Blockmatrix aus (6.13) repräsentiert die Kopplung der Kommunikationsfunktionen Kom2 bis Kom28 zu den grau dargestellten Telegrammen Tel2 bis Tel28, die den kausalen Prozess indirekt beeinflussen können und daher bei der Auslegung eines Monitors mit zu berücksichtigen sind.

Die den in Abbildung 6.4 dargestellten beispielhaften Prozess repräsentierenden Inzidenzmatrizen gemäß der Strukturbeschreibung aus (6.13) und (6.14) können dem Anhang B.3 entnommen werden.

Weiter gelten folgende Zusammenhänge, wobei gemäß der Annahme eines am Kommunikationssystem gekoppelten Monitors gilt, dass eine Stelle als messbar bezeichnet wird, wenn ihr ein eindeutiger Ausgang in Form eines nachfolgenden Kommunikationsprozesses zugewiesen werden kann.

$$\underline{\underline{\boldsymbol{\varphi}}} = \begin{pmatrix} 0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \end{pmatrix} \begin{matrix} \\ \\ \\ \\ \\ \\ \\ \\ \end{matrix} \begin{matrix} \text{Stellen} \\ \\ \\ \\ \\ \\ \\ \\ \text{messbare Stellen} \end{matrix}$$

$$\underline{\underline{\boldsymbol{\varphi}}}\,\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{TF,K,TP,red}} = \begin{pmatrix} 0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,-1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \end{pmatrix} \begin{matrix} \\ \\ \\ \\ \\ \\ \\ \\ \end{matrix} \begin{matrix} \text{Stellen} \\ \\ \\ \\ \\ \\ \\ \\ \text{messbare Stellen} \end{matrix}.$$

Nach den in [Aguirre-Salas/Santoyo-Sanchez 2009] eingeführten Definitionen sind die Transitionen Fkt3 (Spalte 1), Fkt12.1 (Spalte 3), Fkt28 (Spalte 7), Fkt29 (Spalte 8), Fkt30 (Spalte 9), Fkt35 (Spalte 11), Fkt36 (Spalte 12), und Fkt37 (Spalte 13) nicht sichtbar, d. h. keine ihrer Vor- bzw. Nachstellen sind messbar (leere Spalten in $\underline{\underline{\boldsymbol{\varphi}}}\,\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{TF,K,TP,red}}$). Die Spalten der Teilfunktionen Fkt4.1 und Fkt4.2 (Spalte 2 und 22) bzw. Fkt13.1 und Fkt13.2-Fkt13.6 (Spalte 4 und 24) bzw. Fkt16.1 und Fkt16.2 (Spalte 5 und 25) bzw. Fkt31.1 und Fkt31.2-31.6 (Spalte 10 und 26) sind nicht leer aber identisch. Die Eingangsinformationen der den gleichgefüllten Spalten zuzuordnenden Teilfunktionen sind außerdem unbekannt, so dass die einem Spaltenpaar zuzuordnenden Teilfunktionen nach außen nicht voneinander unterscheidbar sind.

Auf Basis der Untersuchungen zur Zustandsdetektierbarkeit werden die nicht sichtbaren bzw. die nach außen nicht unterscheidbare Funktionen bzw. Transitionen zur Untersuchungen der Sequenzdetektierbarkeit in Gruppen zusammengefasst, sofern sie benachbart sind. Die benachbarten Funktionen Fkt3, als nicht sichtbare Funktion und Fkt4.1, als nach außen nicht unterscheidbare Funktion werden der Gruppe Gr1 zugeordnet. Die Funktionen Fkt12.1 und Fkt13.1 bilden die Gruppe Gr2 usw.. Die Gruppen, die benachbarte, nicht sichtbare und nach außen nicht unterscheidbare Transitionen zusammenfassen, sind in Abbildung 6.4 gestrichelt umrandet.

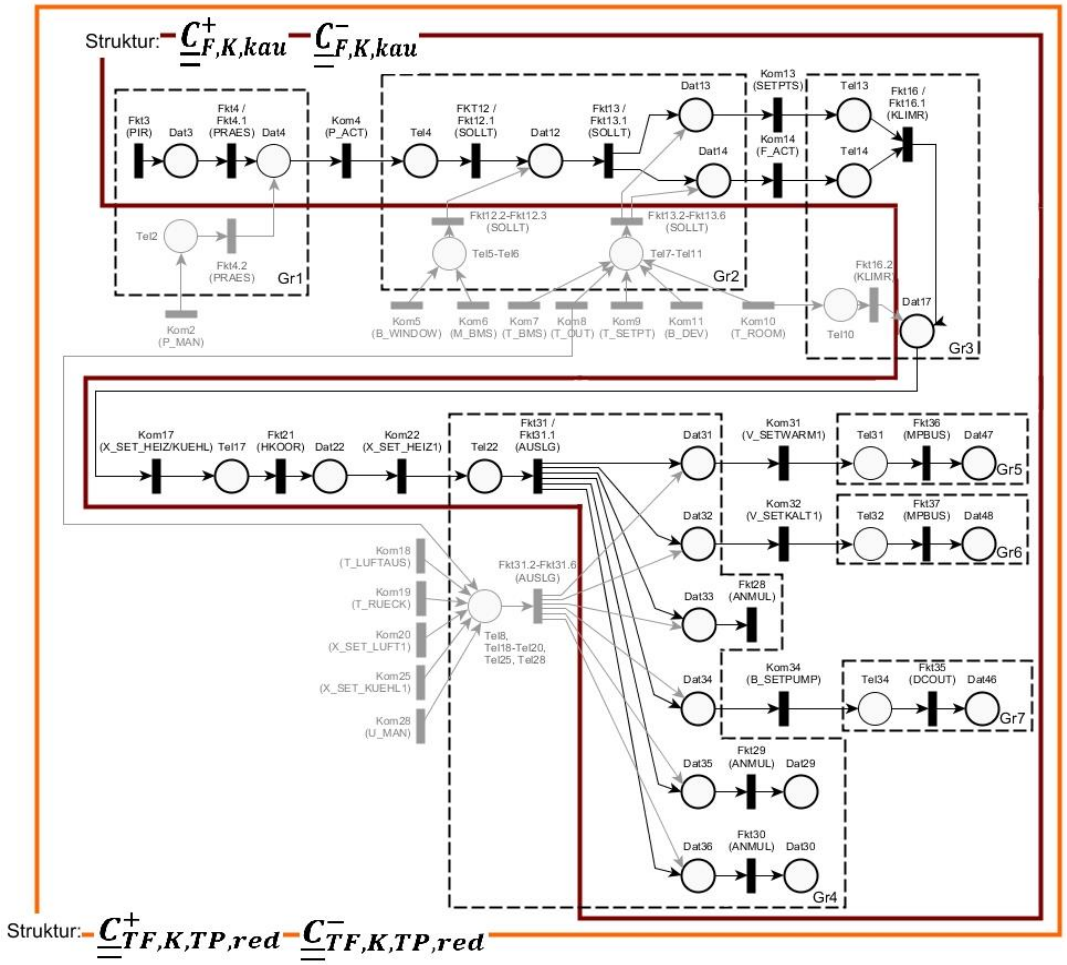


Abbildung 6.4: Petrinetz zur Analyse der Sequenzdetektierbarkeit eines Prozessablaufes mit nicht detektierbaren Zuständen

Lediglich die Gruppen Gr2 und Gr3 erfüllen die eingangs erläuterten Bedingungen für deren Sequenzdetektierbarkeit. Für diese Gruppen kann jeweils garantiert werden, dass die in grau dargestellten Eingangstransitionen der Gruppe nicht schalten, solange sich noch Marken innerhalb der Gruppe befinden, die von der den sequenzzugehörigen Transitionen Kom4 (Eingangstransition von Gruppe Gr2) bzw. Kom13 oder Kom14 (Eingangstransition von Gruppe Gr3) erzeugt wurden.

Für die Gruppe Gr1 können die Bedingungen zur Sequenzdetektierbarkeit nicht eingehalten werden, da ihre Transition Fkt3 jederzeit Konzession aufweist. Die Gruppen Gr4 bis Gr7 erfüllen ebenfalls nicht die Anforderungen an Detektierbarkeit, da die von den Eingangstransitionen erzeugten Marken in den Gruppen verbleiben können, indem sie auf den Stellen Dat29, Dat 30 bzw. Dat47, Dat48 oder Dat46 abgelegt werden.

Demnach ist der betrachtete Prozessablauf nur im Bereich der zusammenhängenden Gruppen Gr2 und Gr3 bis einschließlich der sichtbaren Funktion Fkt21 bzw. deren nachfolgenden Kommunikationsfunktion Kom22 sequenzdetektierbar. Dieser Umstand ist bei der Umsetzung einer Online-Analyse, die in Abschnitt 7.2.2 vorgestellt wird, zu berücksichtigen.

Die Zuordnung relevanter Kommunikationsfunktionen zu den sequenzdetektierbaren Gruppen lässt sich durch folgende Matrizen beschreiben:

$$\underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in}} = \left(\underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in1}} \quad \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in2}}\right) = (gr_{ij}) \text{ mit } gr_{ij} = \begin{cases} 1, & \text{wenn } t_{Kom,j} \in T^E_{Gr,i} \\ 0, & \text{sonst} \end{cases}$$

$$= \begin{pmatrix} 1\,0\,0\,0\,0\,0\,0\,0 & 0\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,1\,1\,0\,0\,0\,0\,0 & 0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0 \end{pmatrix} \begin{matrix} \text{Kommunikationsfunktionen} \\ \text{Sequenzdetektierbare Gruppen} \end{matrix} \tag{6.18}$$

$$\text{mit } \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in1}} = (gr_{ij}) \text{ mit } gr_{ij} = \begin{cases} 1, & \text{wenn Kommunikationsfunktion } j \text{ Eingangstransition der Gruppe } i \text{ ist und im kausalen Ablauf enthalten ist} \\ 0, & \text{sonst} \end{cases}$$

$$\text{mit } \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in2}} = (gr_{ij}) \text{ mit } gr_{ij} = \begin{cases} 1, & \text{wenn Kommunikationsfunktion } j \text{ Eingangstransition der Gruppe } i \text{ ist und nicht im kausalen Ablauf enthalten ist} \\ 0, & \text{sonst} \end{cases}$$

mit $T^E_{Gr,i}$ = Menge aller Eingangstransitionen der Gruppe i

und

$$\underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{out}} = (gr_{ij}) \text{ mit } gr_{ij} = \begin{cases} 1, & \text{wenn } t_{Kom,kau,j} \in T^S_{Gr,i} \\ 0, & \text{sonst} \end{cases}$$

$$= (gr_{ij}) \text{ mit } gr_{ij} = \begin{cases} 1, & \text{wenn Kommunikationsfunktion } j \text{ Ausgangstransition der Gruppe } i \text{ ist und im kausalen Ablauf enthalten ist} \\ 0, & \text{sonst} \end{cases}$$

$$= \begin{pmatrix} 0\,1\,1\,0\,0\,0\,0\,0 \\ 0\,0\,0\,1\,0\,0\,0\,0 \end{pmatrix} \begin{matrix} \text{Kommunikations-funktionen} \\ \text{Sequenzdetektierbare Gruppen} \end{matrix}$$

mit $T^S_{Gr,i}$ = Menge aller Ausgangsstransitionen der Gruppe i. (6.19)

Weiterhin lässt sich die Zuordnung relevanter Kommunikationsfunktionen zu den nach außen sichtbaren Funktionen durch folgende Matrizen darstellen:

$$\underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{in}} = (ts_{ij}) \text{ mit } ts_{ij} = \begin{cases} 1, & \text{wenn } \dot{t}_{Kom,kau,j} = \dot{t}_{Fkt_s,i} \\ 0, & \text{sonst} \end{cases}$$

$$= (ts_{ij}) \text{ mit } ts_{ij} = \begin{cases} 1, & \text{wenn Nachstelle der Kommunikationsfunktion } j \text{ die Vorstelle der sichtb. Funktion } i \\ & \text{ist und Kommunikationsfunktion } j \text{ im kausalen Ablauf enthalten ist} \\ 0, & \text{sonst} \end{cases}$$

$$= \begin{matrix} \text{Kommunikations} - \\ \text{funktionen} \end{matrix} \begin{matrix} \\ (0\ 0\ 0\ 1\ 0\ 0\ 0\ 0) \end{matrix} \begin{matrix} \\ \text{sichtbare Funktion} \end{matrix} \tag{6.20}$$

und

$$\underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{out}} = (ts_{ij}) \text{ mit } ts_{ij} = \begin{cases} 1, & \text{wenn } \dot{t}_{Kom,kau,j} = \dot{t}_{Fkt_s,i} \\ 0, & \text{sonst} \end{cases}$$

$$= (ts_{ij}) \text{ mit } ts_{ij} = \begin{cases} 1, & \text{wenn Vorstelle der Kommunikationsfunktion } j \text{ die Nachstelle der sichtb. Funktion } i \\ & \text{ist und Kommunikationsfunktion } j \text{ im kausalen Ablauf enthalten ist} \\ 0, & \text{sonst} \end{cases}$$

$$= \begin{matrix} \text{Kommunikations} - \\ \text{funktionen} \end{matrix} \begin{matrix} \\ (0\ 0\ 0\ 0\ 1\ 0\ 0\ 0) \end{matrix} \begin{matrix} \\ \text{sichtbare Funktion} \end{matrix} . \tag{6.21}$$

7 Analysen im Online-Betrieb

Während an ein Realsystem gekoppelte Petrinetz-Konstrukte bislang zumeist für die Steuerung und Beobachtung des technischen Prozesses zur Anwendung kamen [Jörns 1997], liegt der Fokus des folgenden Kapitels auf der Nutzung von Petrinetz-Modellen für die Online-Analyse eines verteilten Steuerungsprozesses zu Test- und Diagnosezwecken.

In Abschnitt 7.1 wird eine Umsetzung einer Petrinetzumgebung für eine Online-Analyse vorgestellt. Auf Basis von *farbigen Petrinetzen* [Jensen 1992] wurde zunächst eine Übergabestelle zum Austausch von Telegramminhalten zwischen einem realen, verteilten Steuerungssystem und einer Analyseumgebung realisiert [Diekhake/Schnieder 2013]. Mit der Überführung von Zustandsinformationen aus dem Steuerungsprozess in binäre Petrinetzzustände können so nachfolgenden Analysemodellen einfach auswertbare Zustandsbeschreibungen des Steuerungsprozesses zur Verfügung gestellt werden. In entgegengesetzter Richtung ist aus der Analyseumgebung heraus das Ablegen eines Bustelegramms auf den dem Steuerungsprozess zugehörigen Kommunikationskanal möglich. Die Analysemodelle selbst basieren auf Petrinetzen mit *externen Stellen* [Hanisch 1992] und repräsentieren abhängig von der Analyseanwendung ein Stimulations- und Auswertesystem.

Die Anwendungen von Petrinetzen für die Online-Analyse eines Steuerungsprozesses sind vielfältig. In [Bachmann 2013] sind beispielsweise Petrinetze für Testausführungen behandelt worden. Im Vergleich zu anderen Beschreibungssprachen, die einen ausführbaren Testfalls abbilden, hat sich gezeigt, dass mit Hilfe von Petrinetzmodellen die einzelnen Schritte der Testfallimplementierung transparent darstellt werden können und eine eindeutige Unterscheidung zwischen einer Testeingabe und einem erwarteten Ergebnis vorgenommen werden kann. Die Abbildung eines konkreten Testfallablaufes in einem Analysemodell wird in Abschnitt 7.2.1 näher behandelt und für das Beispiel der Funktion zur Belegungsauswertung aus Abschnitt 1.1.1 vorgestellt. In [Diekhake/Schnieder 2013] und [Zhu 2014] wird als weitere Anwendung der Online-Analyse mit Petrinetzen das Monitoring eines Steuerungsprozesses zur Laufzeitüberwachung beschrieben. Mit Hilfe einer Laufzeitüberwachung können Verletzungen temporaler Eigenschaften in Echtzeit diagnostiziert und so schnellstmöglich vor funktionalem Fehlverhalten innerhalb des verteilten Automatisierungssystems gewarnt werden. In Abschnitt 7.2.2 wird daher ein Analysemodell vorgestellt, welches den Prozess zur Prüfung der Einhaltung von Zeitschranken am Beispiel des in Abschnitt 5.3 eingeführten Funktionsablaufes abbildet.

Um für die Online-Analyse eine Wiederverwendung von vorhandenen Strukturinformationen herbeizuführen [vgl. Berger et al. 2012] und um eine automatisierte Modellerstellung zu unterstützen, werden das Vorwissen in einem Systemmodell abgelegt und die Strukturen der Analysemodelle mathematisch formuliert. Als Online-Analysemodelle werden beispielhaft ein Testausführungsprozess und ein Prozess zur Laufzeitüberwachung behandelt.

7.1 Umsetzung einer Petrinetzumgebung zur Online-Analyse

Die umgesetzte Petrinetzumgebung zur Online-Analyse besteht u. a. aus einer Telegrammübergabe-Ebene (Abschnitt 7.1.1), die einen Zugriff der Analysemodelle auf das Kommunikationssystem des Automatisierungssystems erlaubt. Desweiteren sind die Analysemodelle selbst umzusetzen, d. h. diejenigen Petrinetze, die ein der Analyseaufgabe entsprechendes (Stimulations- und) Auswertesystem repräsentieren (Abschnitt 7.1.2).

7.1.1 Telegrammübergabe

Die Telegrammübergabe-Ebene stellt das Bindeglied zwischen dem zu analysierenden Steuerungssystem und den Analysemodellen dar. Die Kommunikation zwischen der Telegrammübergabe-Ebene und dem Automatisierungssystem erfolgt via LAN über das TCP-Protokoll. Die Vermittlung der Daten wird über eine CORBA-Implementierung koordiniert, für die folgende Dienste bereitgestellt sind:

- *void allTransitionsWithThisNameFired (in string transitionName, in string value);*
- *void MessageWithThisNameSend (in string messageName, in string messageValue);*

Jedes vom Automatisierungssystem auf den Bus abgelegte Telegramm löst den Aufruf der CORBA-Funktion *allTransitionsWithThisNameFired* aus, welche eine entsprechende Transition auf der Ebene der Telegrammübergabe aktiviert und damit zu deren Schalten anregt. Nach dem Feuern dieser Transition wird das Telegramm seitens des Monitors als empfangen markiert. Es folgt die Auswertung des übergebenen Telegramminhaltes, der in einer farbigen Marke abgelegt wurde. Für eine vereinfachte weiterführende Behandlung wird der Inhalt logisch zugeordnet, d. h. binär ausgewertet. Über die binäre Zuordnung erfolgt die Weiterleitung der Zustandsinformationen von der Telegrammübergabe-Ebene zu den übergeordneten Analysemodellen, die damit lediglich mit einfach zu beschreibenden Zuständen arbeiten. In Gegenrichtung kann eine Transition aus dem Petrinetz die CORBA-Funktion *MessageWithThisNameSend*

ausführen, wodurch ein Telegramm auf den Kommunikationsbus des Automatisierungssystems abgelegt wird.

7.1.2 Analysemodelle

Die Analysemodelle basieren auf Petrinetzen mit externen Stellen, wobei die externen Stellen das Ein- und Auskoppeln von Informationen ermöglichen. Im Falle des Einlesens einer binär ausgewerteten Information, die aus der Telegrammübergabe-Ebene geliefert wird, wird eine Marke auf die entsprechende externe Stelle des Analysemodells ablegt, die anschließend von diesem konsumiert wird [Jörns 1997: 40]. In Gegenrichtung erfolgt eine Initiierung eines Telegrammversandes, durch das Markieren einer externen Stelle aus dem Analysemodell heraus.

7.1.3 Gekoppelte Analyseumgebung

In Abbildung 7.1 ist der Gesamtprozess vom Empfang eines Bus-Telegrammes über dessen binäre Auswertung bis zur Schaltbeeinflussung einer Transition im Analysemodell dargestellt. Nach der Aktivierung der Transition *Name* wird das zugehörige Telegramm als empfangen markiert. Anschließend erfolgt die Auswertung des Telegramminhaltes. Dieser wird gemäß hinterlegter Bedingungen binär ausgewertet und über die entsprechende externe Stelle im Analysemodell übergeben.

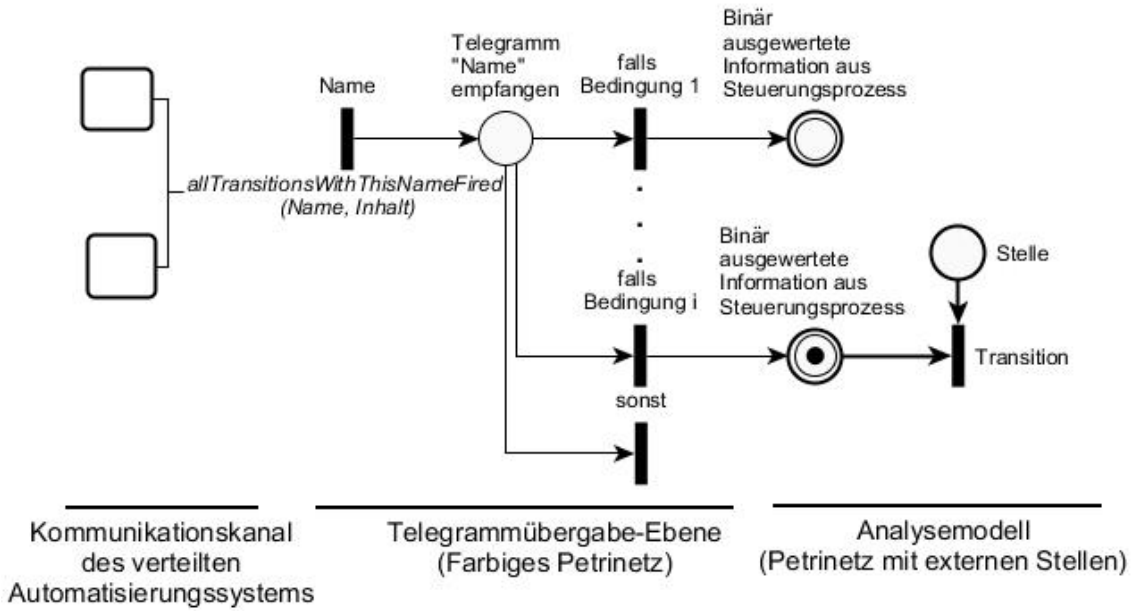


Abbildung 7.1: Einlesen und binäres Auswerten eines Telegrammes zur Schaltbeeinflussung von Transitionen eines Petrinetzes mit externen Stellen

Sind innerhalb des Automatisierungssystems bestimmte Systemzustände herbeizuführen (z. B. Initialisierung eines Testfalles) oder einzelne Ereignisse zu simulieren (z. B. Ausführung einer Testeingabe), ist ein Zugriff des Analysemodells auf das Realsystem notwendig. Der dafür notwendige Prozess einer Telegrammaussendung auf dem Kommunikationskanal des Automatisierungssystems ist in Abbildung 7.2 dargestellt. Sobald die externe Stelle mit einer Marke belegt wird, wird dieser Stelle die Marke über die Telegrammübergabe-Ebene anschließend wieder entzogen, um einen Telegrammversand einzuleiten. D. h. initiiert durch eine markierte externe Stelle im Analysemodell, ausgelöst durch das Schalten einer nachfolgend gekoppelten Transition aus der Telegrammübergabe-Ebene und umgesetzt durch die Ausführung der CORBA-Funktion *MessageWithThisNameSend* wird ein Telegramm auf den Kommunikationskanal des verteilten Automatisierungssystems abgelegt.

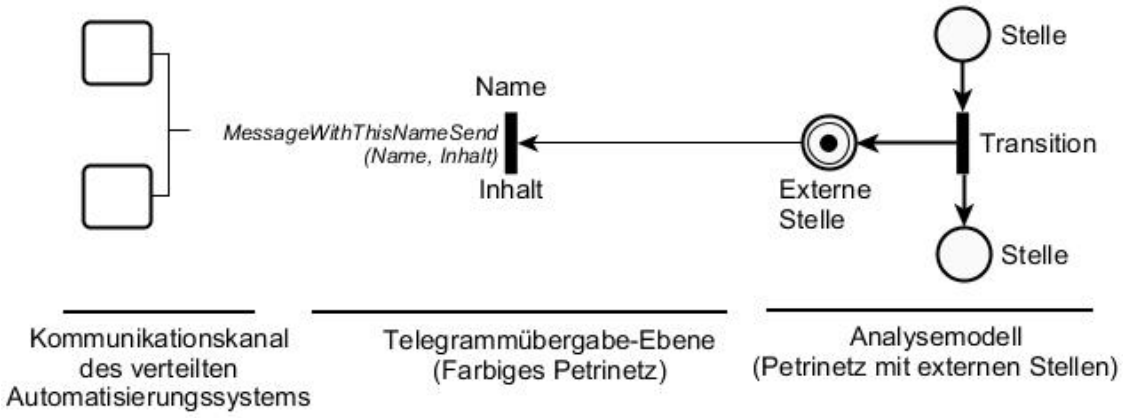


Abbildung 7.2: Externe Stelle mit zugewiesenem Funktionsaufruf zur Telegrammaussendung

7.2 Modellbeispiele für die Online-Analyse

Im Folgenden werden als Analysemodelle die Prozesse einer Testfalldurchführung (Abschnitt 7.2.1) und einer Laufzeitauswertung (Abschnitt 7.2.2) als Anwendungsmöglichkeiten der Online-Analyse allgemein formuliert und für die Beispielsysteme aus Abschnitt 1.1.1 näher erörtert. Das diesem Abschnitt zugrundeliegende Systemmodell ist in Abbildung 7.3 dargestellt. Es stellt für die zu analysierenden Teilsysteme (*Belegungsauswertung* und *kausaler Prozess* aus Abschnitt 5.3) die jeweiligen Anforderungen (*Richtigkeit* und *Zeitinvarianz*) dar und beschreibt die den Analysemodellen zuzuordnenden Eigenschaften.

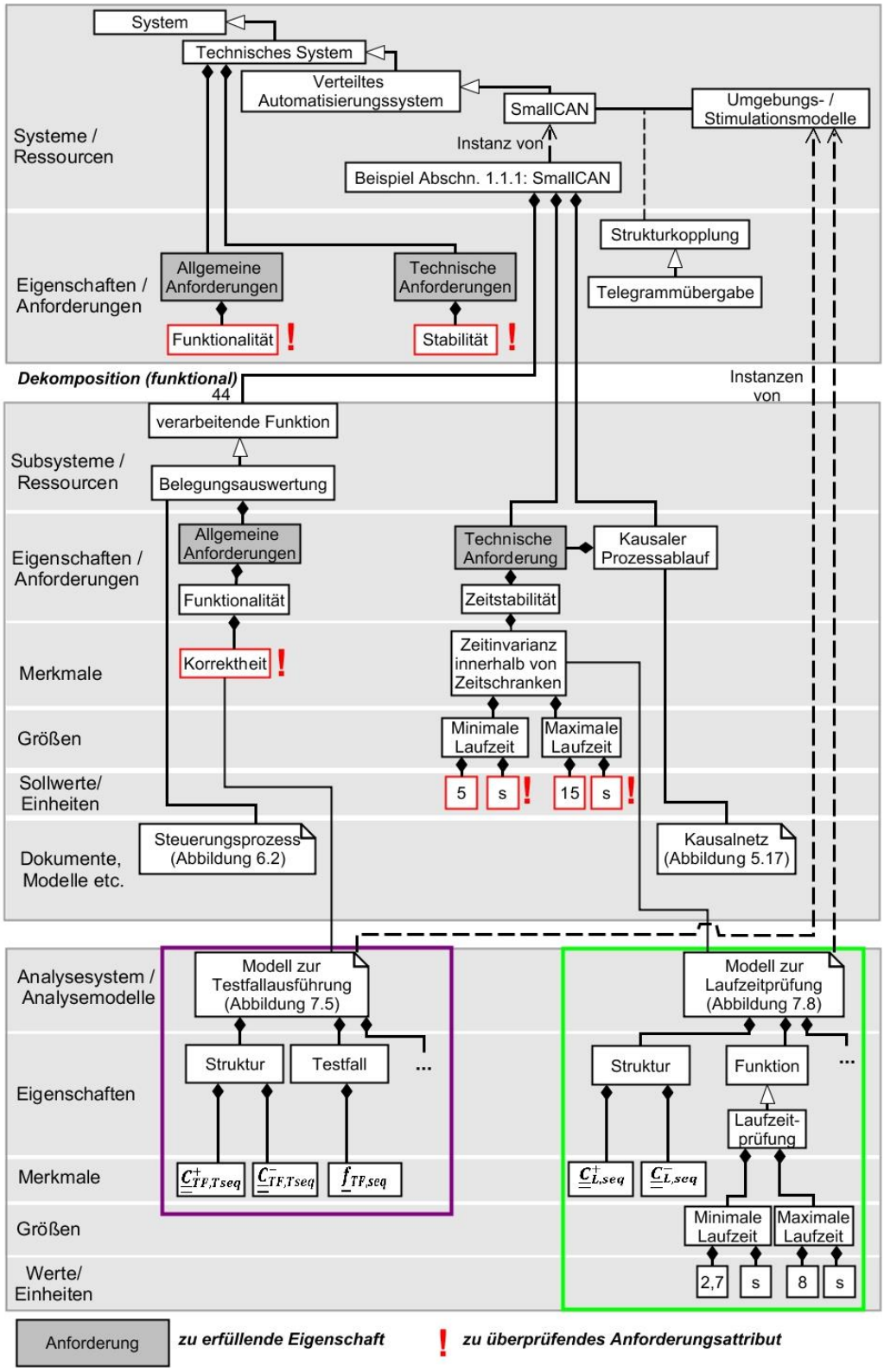


Abbildung 7.3: Systemmodell zur Darstellung der in diesem Abschnitt herangezogenen Analysemodelle für die Online-Analyse am Beispiel einer Testausführung (lila) und einer Laufzeitprüfung (hellgrün)

7.2.1 Modell zur Ausführung eines Testprozesses

In Abbildung 7.4 wird auf Basis eines instrumentierten und ausführbaren Petrinetzes der Prozess zur Durchführung eines implementierten Testfalles dargestellt. Der Ablauf beinhaltet zunächst die Zuweisung nötiger Ausgaben, um das zu testende System in den gültigen Initialzustand zu überführen. Dadurch werden die Vorbedingungen zur Durchführung der Testschritte hergestellt. Die Testschritte setzen sich aus der Ausführung von Prüfanweisungen bzw. Testeingaben sowie den anschließenden Ergebnisbeurteilungen zusammen. Sofern sämtliche erwartete Nachbedingungen erfüllt werden, wird die Stelle *Testende* markiert, die einen bestandenen Testfall repräsentiert.

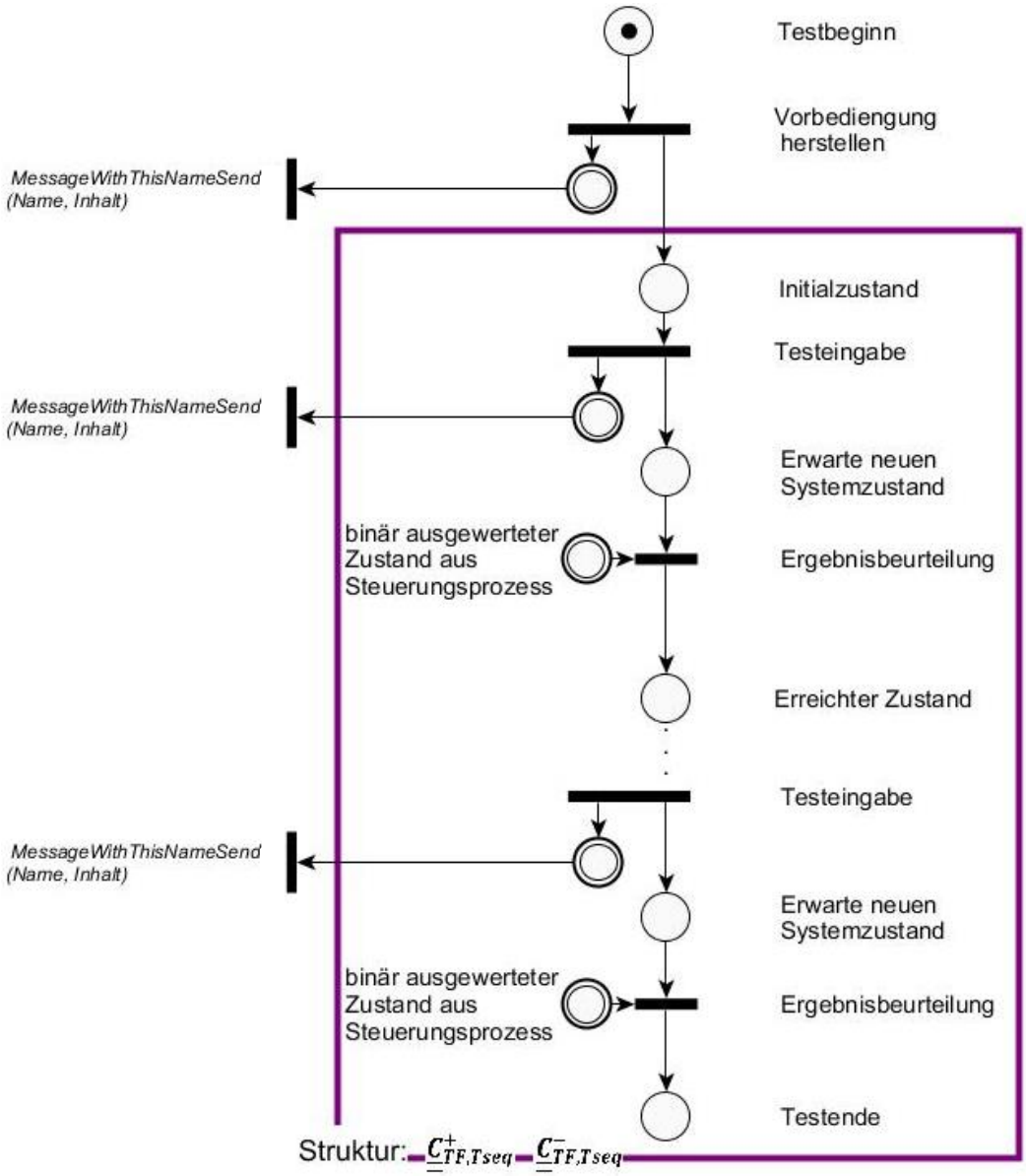


Abbildung 7.4: Petrinetzmodell zur Ausführung eines Testfalles auf Basis des Modellkonstruktes eines Testablaufes (lila)

Der in Abbildung 7.4 aufgezeigte Testfall basiert auf einem definierten kausalen Pfad innerhalb des zu testenden Systems von einer Anfangsmarkierung ausgehend, über definierte Zwischenmarkierungen bis zum Erreichen einer Zielmarkierung, die das Testende darstellt. Die am Testfall beteiligten (Teil-)Funktionen bzw. Daten des zu testenden Systems werden über folgende Vektoren beschrieben:

$$\underline{\boldsymbol{f}}_{\boldsymbol{TF,seq}} = (f_i)\ mit\ f_i = \begin{cases} 1, & \text{wenn Teilfunktion } i \text{ im Pfad vorhanden ist} \\ 0, & \text{sonst} \end{cases} \tag{7.1}$$

bzw.

$$\underline{\boldsymbol{d}}_{\boldsymbol{TF,seq}} = (d_j) = (\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{TF}}\ \underline{\boldsymbol{f}}_{\boldsymbol{TF,seq}})^{\mathrm{T}} \cup (\underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{TF}}\ \underline{\boldsymbol{f}}_{\boldsymbol{TF,seq}})^{\mathrm{T}}\ . \tag{7.2}$$

Demnach lässt sich die Struktur zur Darstellung des Testpfades wie folgt ermitteln:

$$\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{TF,kau}} = (c_{ij})\ \text{mit}\ c_{ij} = \begin{cases} 1, & \text{wenn } \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{TF},ij}=1\ \wedge\ \underline{\boldsymbol{f}}_{\boldsymbol{TF,seq},j}=1 \wedge \underline{\boldsymbol{d}}_{\boldsymbol{TF,seq},i}=1 \\ 0, & \text{sonst} \end{cases} \tag{7.3}$$

und

$$\underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{TF,kau}} = (c_{ij})\ \text{mit}\ c_{ij} = \begin{cases} 1, & \text{wenn } \underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{TF},ij}=1\ \wedge\ \underline{\boldsymbol{f}}_{\boldsymbol{TF,seq},j}=1\ \wedge\ \underline{\boldsymbol{d}}_{\boldsymbol{TF,seq},i}=1 \\ 0, & \text{sonst} \end{cases} \tag{7.4}$$

bzw. nach der Löschung leerer, irrelevanter Zeilen und Spalten in einer reduzierten Darstellung:

$$\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{TF,red}} = \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{TF,kau}}\ \text{mit}\ (c_i) = \begin{cases} \emptyset, & \text{wenn } \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{TF,kau},i}=\underline{\boldsymbol{0}}\ \wedge \underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{TF,kau},i}=\underline{\boldsymbol{0}} \\ (c_i), & \text{sonst} \end{cases}$$

$$\text{mit}\ (c_j) = \begin{cases} \emptyset, & \text{wenn } \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{TF,kau},j}=\underline{\boldsymbol{0}}\ \wedge \underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{TF,kau},j}=\underline{\boldsymbol{0}} \\ (c_j), & \text{sonst} \end{cases} \tag{7.5}$$

$$\underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{TF,red}} = \underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{TF,kau}}\ \text{mit}\ (c_i) = \begin{cases} \emptyset, & \text{wenn } \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{TF,kau},i}=\underline{\boldsymbol{0}}\ \wedge \underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{TF,kau},i}=\underline{\boldsymbol{0}} \\ (c_i), & \text{sonst} \end{cases}$$

$$\text{mit}\ (c_j) = \begin{cases} \emptyset, & \text{wenn } \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{TF,kau},j}=\underline{\boldsymbol{0}}\ \wedge \underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{TF,kau},j}=\underline{\boldsymbol{0}} \\ (c_j), & \text{sonst} \end{cases}\ . \tag{7.6}$$

Der in Abbildung 7.4 dargestellte Prozess zur Ausführung eines Testfalles lässt sich somit nach folgendem Schema aufbauen:

$$\underline{\underline{C}}^{+}_{TF,Tseq} = \begin{pmatrix} \text{Testeingabe-prozesse} & \text{Beurteilungs-prozesse} & \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{C}}_{12} & \text{Erreichte Zustände} \\ \underline{\underline{C}}_{21} & \underline{\underline{\mathbf{0}}} & \text{Erwartete Zustände} \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} & \text{Ext. Stellen (Eingänge)} \\ \underline{\underline{C}}_{41} & \underline{\underline{\mathbf{0}}} & \text{Ext. Stellen (Ausgänge)} \end{pmatrix}$$

$$= \begin{pmatrix} \underline{\underline{\mathbf{0}}} & \underline{\underline{C}}^{+}_{TF,red} \\ \left(\underline{\underline{C}}^{+}_{TF,red}{}^{\mathrm{T}}\,\underline{\underline{C}}^{+}_{TF,red}\right) .* \underline{\underline{I}} & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{C}}_{41} & \underline{\underline{\mathbf{0}}} \end{pmatrix} \tag{7.7}$$

und

$$\underline{\underline{C}}^{-}_{TF,Tseq} = \begin{pmatrix} \text{Testeingabe-prozesse} & \text{Beurteilungs-prozesse} & \\ \underline{\underline{C}}_{11} & \underline{\underline{\mathbf{0}}} & \text{Erreichte Zustände} \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{C}}_{22} & \text{Erwartete Zustände} \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{C}}_{32} & \text{Ext. Stellen (Eingänge)} \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} & \text{Ext. Stellen (Ausgänge)} \end{pmatrix}$$

$$= \begin{pmatrix} \underline{\underline{C}}^{-}_{TF,red} & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{\mathbf{0}}} & \left(\underline{\underline{C}}^{-}_{TF,red}{}^{\mathrm{T}}\,\underline{\underline{C}}^{-}_{TF,red}\right) .* \underline{\underline{I}} \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{C}}_{32} \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} \end{pmatrix} \tag{7.8}$$

mit $\underline{\underline{C}}_{41} = \underline{\underline{E}}_{TF}$ mit $c_{ij} = \begin{cases} 1, \text{ wenn } \underline{E}_{TF,ij}{=}1 \wedge \underline{f}_{TF,seq,i}{=}1 \wedge \underline{d}_{TF,seq,j}{=}1 \\ 0, \text{ sonst} \end{cases}$

$$\text{mit } (c_i) = \begin{cases} \emptyset, \ wenn\ \underline{\underline{C}}_{41,i}{=}\underline{\underline{\mathbf{0}}} \\ (c_i), \ sonst \end{cases}$$

$$\text{mit } (c_j) = \begin{cases} \emptyset, \text{ wenn } \underline{\underline{C}}^{+}_{TF,kau,j}{=}\underline{\underline{\mathbf{0}}} \wedge \underline{\underline{C}}^{-}_{TF,kau,j}{=}\underline{\underline{\mathbf{0}}} \\ (c_j), \text{ sonst} \end{cases} \tag{7.9}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{32}} = \underline{\underline{\boldsymbol{S}}}_{\boldsymbol{TF}}\ \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{TF,red}} \text{ mit } c_j = \begin{cases} 1, \text{ wenn } \underline{\underline{\boldsymbol{S}}}_{\boldsymbol{TF},ij}\ \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{TF,red},ij} = 1 \wedge \underline{\boldsymbol{f}}_{\boldsymbol{TF,seq},i} = 1 \wedge \underline{\boldsymbol{d}}_{\boldsymbol{TF,seq},j} = 1 \\ 0, \text{ sonst} \end{cases}$$

$$\text{mit } (c_i) = \begin{cases} \emptyset, \text{ wenn } \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{32},i} = \underline{\boldsymbol{0}} \\ (c_i), \text{ sonst} \end{cases}$$

$$\text{mit } (c_j) = \begin{cases} \emptyset, \text{ wenn } \underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{TF,kau},j} = \underline{\boldsymbol{0}} \wedge \underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{TF,kau},j} = \underline{\boldsymbol{0}} \\ (c_j), \text{ sonst} \end{cases} . \qquad (7.10)$$

Die ersten beiden Zeilen der Blockmatrizen (7.7) und (7.8) stellen die eigentliche Struktur des Testfalles dar. Jeder (Teil-)Funktion des Pfades aus (7.3) wird ein Prozess zur Testeingabe sowie ein Prozess zur Beurteilung zugewiesen. Die Testeingabe- und Beurteilungsschritte werden über die erreichten und erwarteten Zustände miteinander gekoppelt. Die Transitionen der Testeingabe initiieren eine Stimulation des Steuerungssystems. Die Auswertung der Beurteilungsschritte erfolgt auf Basis rückgemeldeter Zustände. Die Verknüpfungen an das Steuerungssystem erfolgt mit der Anbindung der Testeingabeprozesse an die externen Ausgänge und mit der Anbindung der Beurteilungsprozesse an die externen Eingänge. Diese Kopplungen werden in den letzten beiden Zeilen der Blockmatrizen aus (7.7) und (7.8) abgebildet.

Anwendungsbeispiel

Sofern die in Abschnitt 6.2 vorgestellte Kontrollstruktur realisierbar ist (Steuerbarkeit der Verarbeitungsfunktion zur Belegungsauswertung), lässt sich der in Abbildung 7.5 dargestellte Prozess eines Testdurchlaufes ausführen. Der Testfall basiert auf einem definierten Pfad vom Zustand *keine Präsenz* über die *kurze Präsenz* bis zum Erreichen des Zustandes der *langen Präsenz*. Folgende Transitionen (Teilfunktionen) bzw. internen Daten kommen in dem festgelegten Funktionsablauf vor:

$$\underline{\boldsymbol{f}}_{\boldsymbol{TF,seq}}{}^{\mathrm{T}} = (1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0)$$

und

$$\underline{\boldsymbol{d}}_{\boldsymbol{TF,seq}} = (d_j) = (1\,1\,1\,0\,0)$$

Aus dem Initialzustand heraus wird über das Testsystem zunächst ein Bewegungsalarm P_AUTO-1 sowie ein Präsenzzustand P_AUTO-0 simuliert. Nach dieser Testeingabe wird vom Steuerungsprozess die Meldung der kurzen Präsenz P_ACT-1 erwartet. Je nachdem ob sich der Steuerungsprozess aktuell in der kurzen Präsenz mit bzw. ohne ungeblockte Ablaufwarnung befindet (M_1 bzw. M_3), wird nach Ablauf von Δt3, d. h. der Scharfschaltung der langen Präsenz, ein Präsenzzustand P_AUTO-0 bzw. ein Bewegungsalarm P_AUTO-1 über das Testsystem simuliert. Sobald der

Steuerungsprozess die lange Präsenz mit P_ACT-2 quittiert, gilt der Testfall als bestanden.

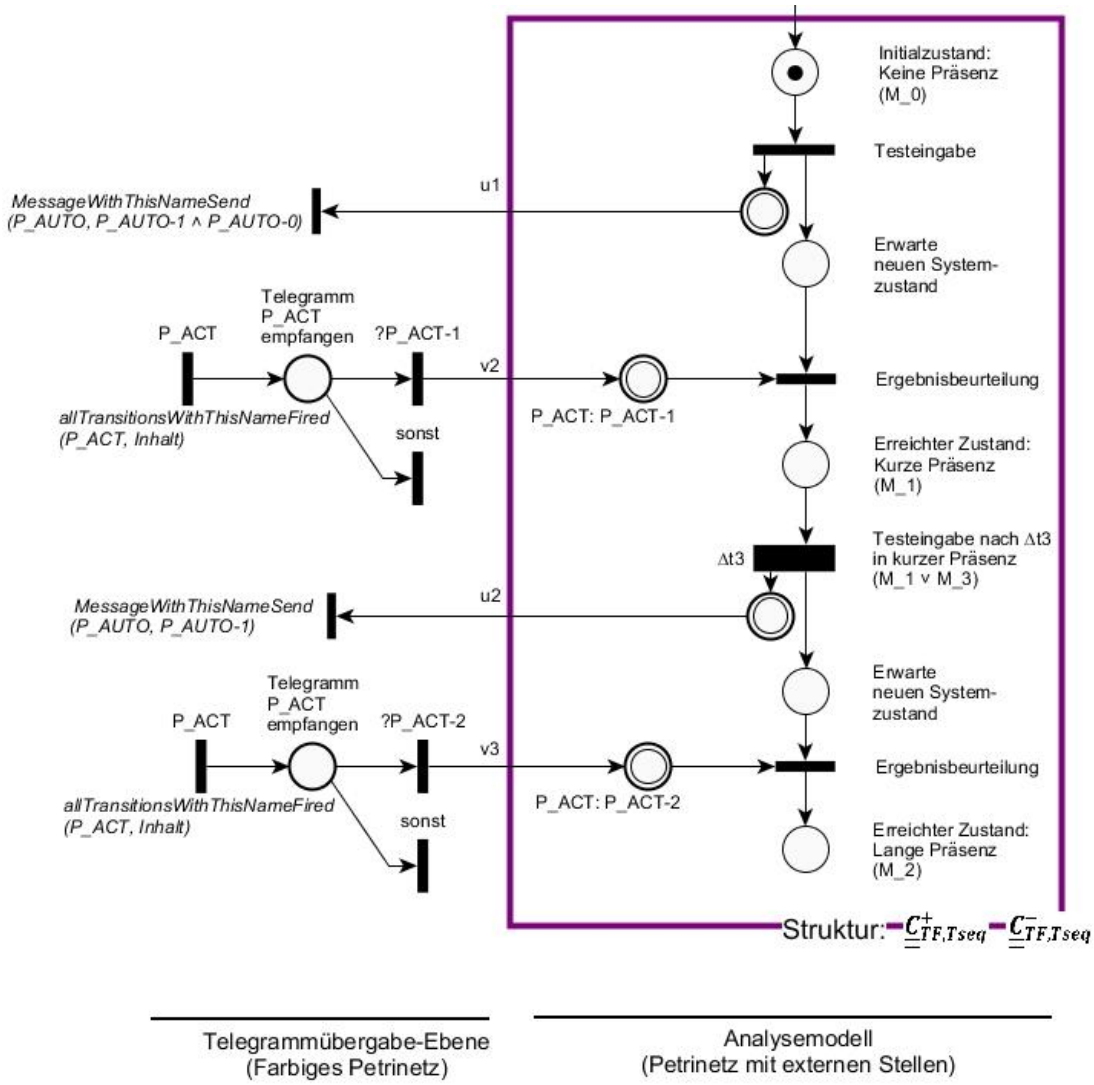


Abbildung 7.5: Prozess für die Durchführung eines exemplarischen Testfalls für die Funktion der Belegungsauswertung

Die Struktur zur Darstellung des Pfades stellen folgende Matrizen dar:

$$\underline{\underline{C}}^{+}_{TF,kau} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{matrix} \text{Teilfunktionen} \\ \\ \\ \\ \\ \text{Daten} \end{matrix}$$

bzw. $$\underline{\underline{C}}^{+}_{TF,red} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{matrix} \text{Teilfunktionen} \\ \\ \text{Daten} \end{matrix}$$

und

$$\underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{TF,kau}} = \begin{pmatrix} 1&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0 \end{pmatrix} \begin{matrix} \\ \\ \\ \\ \\ \end{matrix} \begin{matrix} \text{Teilfunktionen} \\ \\ \\ \\ \\ \text{Daten} \end{matrix}$$

bzw. $\underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{TF,red}} = \begin{pmatrix} 1&0 \\ 0&1 \\ 0&0 \end{pmatrix}$ $\begin{matrix} \text{Teilfunktionen} \\ \\ \\ \text{Daten} \end{matrix}$

Die Struktur des Analysemodells (Abbildung 7.5, lila), das für die Durchführung des Testablaufes *keine Präsenz → kurze Präsenz → langen Präsenz* zur Anwendung kommt, lässt sich entsprechend der Matrizen (7.7) und (7.8) wie folgt beschreiben:

$$\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{TF,Tseq}} = \begin{pmatrix} \underline{\underline{\mathbf{0}}} & \begin{pmatrix} 0&0 \\ 1&0 \\ 0&1 \end{pmatrix} \\ \begin{pmatrix} 1&0 \\ 0&1 \end{pmatrix} & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} \\ \begin{pmatrix} 1&0 \\ 0&1 \end{pmatrix} & \underline{\underline{\mathbf{0}}} \end{pmatrix} \begin{matrix} \\ \text{Erreichte Zustände} \\ \text{Erwartete Zustände} \\ \text{Ext. Stellen (Eingänge)} \\ \text{Ext. Stellen (Ausgänge)} \end{matrix}$$

Testeingabe-prozesse: Spalte 1; Beurteilungs-prozesse: Spalte 2

und

$$\underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{TF,Tseq}} = \begin{pmatrix} \begin{pmatrix} 1&0 \\ 0&1 \\ 0&0 \end{pmatrix} & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{\mathbf{0}}} & \begin{pmatrix} 1&0 \\ 0&1 \end{pmatrix} \\ \underline{\underline{\mathbf{0}}} & \begin{pmatrix} 1&0 \\ 0&1 \end{pmatrix} \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} \end{pmatrix} \begin{matrix} \\ \text{Erreichte Zustände} \\ \text{Erwartete Zustände} \\ \text{Ext. Stellen (Eingänge)} \\ \text{Ext. Stellen (Ausgänge)} \end{matrix} .$$

Testeingabe-prozesse: Spalte 1; Beurteilungs-prozesse: Spalte 2

7.2.2 Modell zur Laufzeitüberwachung

Die Überwachung der Laufzeit eines Steuerungsprozesses während des Betriebes lässt sich über das Petrinetz aus Abbildung 7.6 realisieren. Nach dem Eintreten des initiierenden Zustands, ausgelesen aus dem realen Steuerungsprozesses, wird der erste Prozessschritt innerhalb des Analysemodells ausgeführt und damit die Laufzeitüberwachung, die im rechten Ast in Abbildung 7.6 modelliert ist, gestartet. Nach Ablauf der minimal erwarteten Laufzeit $t_{seq,min}$ für den Steuerungsprozess wird die zeitbehaftete Transition des ersten Zeitgliedes geschaltet. Ist die maximal erwartete Laufzeit $t_{seq,max}$ erreicht (Abschnitt 5.3), schaltet auch die nachfolgende zeitbehaftete Transition.

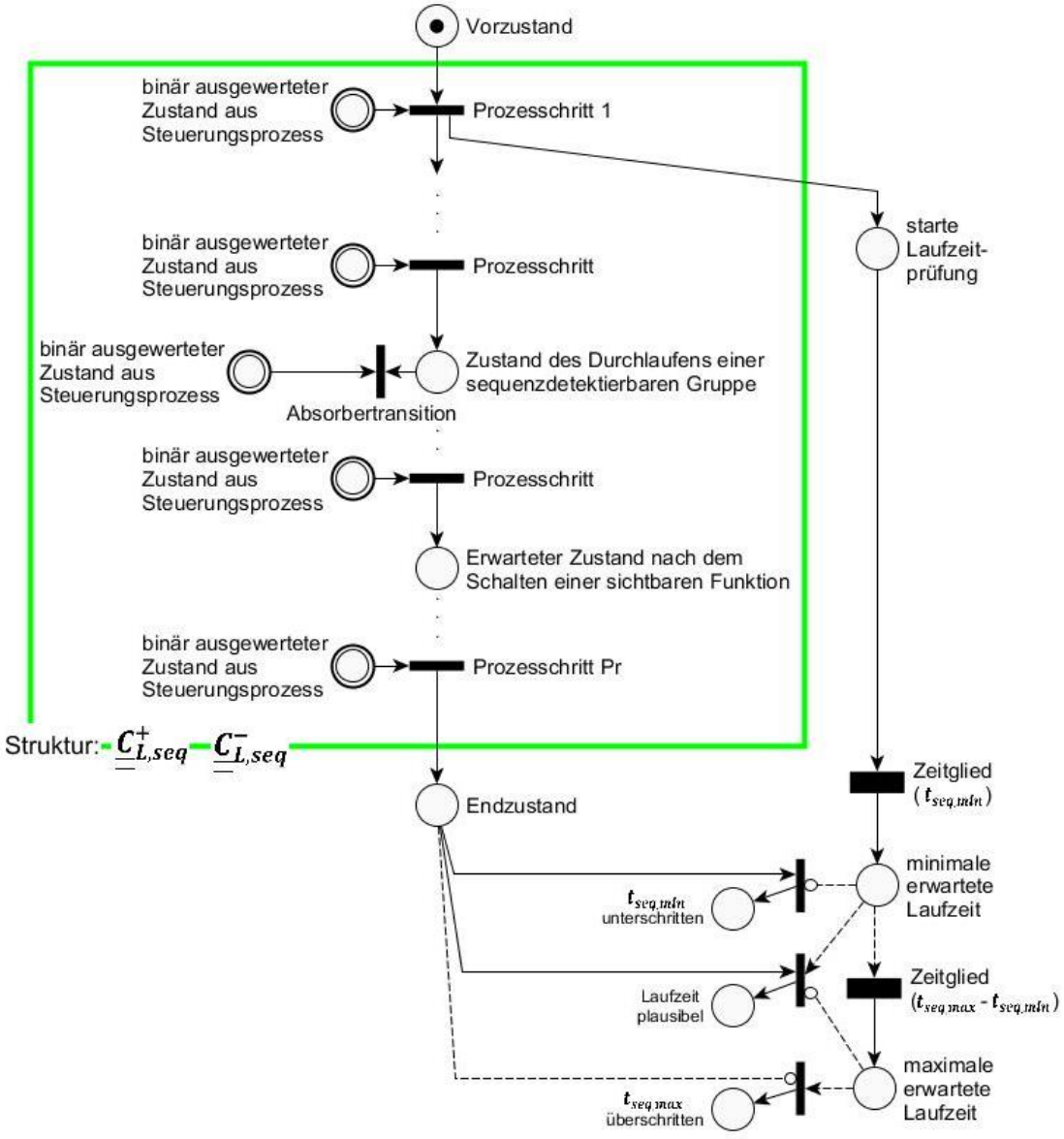


Abbildung 7.6: Petrinetzmodell zur Laufzeitprüfung eines Steuerungsprozesses auf Basis des Modellkonstruktes (hellgrün) zur Erfassung des Realverhaltens des Steuerungssystems

Folgende drei Szenarien können demnach betrachtet werden:

Szenario 1

> Sofern die minimale Zeitschranke noch nicht markiert ist, der Steuerungsprozess aber bereits seinen Endzustand erreicht hat, liegt eine nicht-plausible Laufzeit vor. Die minimale Laufzeit wurde unterschritten. Als mögliche Ursache kann eine fälschliche Zuordnung unabhängiger Ereignisse in die kausale Abfolge angenommen werden. Eine solche fälschliche Zuweisung kann beispielsweise in Systemen auftreten, in denen keine *chronologische Totalordnung* vorliegt, also insbesondere in nicht-synchronisierten Systemen mit parallel ausgeführten Steuerungsprozessen (vgl. Parallelität in Abschnitt 1.2).

Szenario 2

> Sind sowohl die Stelle zur Repräsentation der minimal zu erwartenden Laufzeit als auch die Stelle zur Repräsentation der maximal zu erwartenden Laufzeit je mit einer Marke belegt, der Steuerungsprozess jedoch noch nicht durchlaufen, liegt eine nicht-plausible Verzögerung vor, d. h. die maximale Laufzeit wurde überschritten. Sofern es sich bei der beobachteten Abfolge um die erwartete kausal zusammenhängende Ereigniskette handelt, kann u. a. ein (Teil-)Ausfall im System als Ursache angenommen werden.

Szenario 3

> Im Falle eines abgeschlossenen Steuerungsprozesses nach Eintritt der minimalen Zeitschranke und vor Eintritt der maximalen Zeitschranke weist die Laufzeit einen plausiblen Wert auf. Die Ereigniskette ist innerhalb der erwarteten Zeitschranken beobachtet worden.

Die der Laufzeitanalyse heranzuziehenden Prozessschritte, die über die externen Eingangsstellen entsprechend dem Realverhalten des Steuerungsprozesses aktiviert werden, sind in Abbildung 7.6 im linken Ast dargestellt. Die abzubildenden Prozessschritte sind entsprechend den Ergebnissen aus der Beobachtbarkeitsanalyse (Abschnitt 6.3), d. h. auf Basis der sequenzdetektierbaren Gruppen und sichtbarer Funktionen, zu bestimmen um eine fälschliche Zuweisung von Ereignissen (Szenario 1) zu vermeiden Durch einen direkten Abgleich des beobachteten Realverhaltens mit den im Modell parametrierten Grenzwerten gemäß Abschnitt 5.3 lässt sich so die Laufzeit des Prozessablaufes auf Plausibilität überprüfen. Die Struktur des hellgrün umrahmten Petrinetzmodells aus Abbildung 7.6 baut auf den folgenden mathematischen

Konstrukten auf, welche mit Hilfe der eingeführten Matrizen (6.18), (6.19), (7.24) und (7.25) erstellt werden können und im Folgenden erläutert werden:

$$\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{L,seq}} = \begin{pmatrix} \begin{pmatrix} \underline{\underline{\boldsymbol{C}}}_{\mathbf{11}} & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{\boldsymbol{C}}}_{\mathbf{22}} \end{pmatrix} & \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} \end{pmatrix} \begin{matrix} \\ \\ \end{matrix}$$

$Pr-1$ Prozessschritte | Prozessschritt Pr | Absorber-transitionen

Zustand des Durchlaufens einer sequenzdetektierbaren Gruppe

Erwarteter Zustand nach dem Schalten einer sichtbaren Funktion

Ext. Stellen (Eingänge)

$$= \begin{pmatrix} \begin{pmatrix} \left(\underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in1}}\ \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in1}}{}^{\mathrm{T}}\right) .* \underline{\boldsymbol{I}} & \underline{\mathbf{0}} \\ \underline{\mathbf{0}} & \left(\underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{in}}\ \underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{in}}{}^{\mathrm{T}}\right) .* \underline{\boldsymbol{I}} \end{pmatrix} & \underline{\mathbf{0}} & \underline{\mathbf{0}} \\ \underline{\mathbf{0}} & \underline{\mathbf{0}} & \underline{\mathbf{0}} \end{pmatrix} \tag{7.11}$$

und

$$\underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{L,seq}} = \begin{pmatrix} \begin{pmatrix} \underline{\underline{\boldsymbol{C}}}_{\mathbf{11}} & \underline{\underline{\boldsymbol{C}}}_{\mathbf{12}} \\ \underline{\underline{\boldsymbol{C}}}_{\mathbf{21}} & \underline{\underline{\boldsymbol{C}}}_{\mathbf{22}} \end{pmatrix} & \underline{\boldsymbol{C}}_{\mathbf{13}} & \underline{\underline{\boldsymbol{C}}}_{\mathbf{14}} \\ \underline{\underline{\boldsymbol{C}}}_{\mathbf{31}} & \underline{\underline{\boldsymbol{C}}}_{\mathbf{32}} & \underline{\boldsymbol{C}}_{\mathbf{33}} & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} & \underline{\mathbf{0}} & \underline{\underline{\boldsymbol{C}}}_{\mathbf{34}} \end{pmatrix} \begin{matrix} \\ \\ \\ \end{matrix} \tag{7.12}$$

$Pr-1$ Prozessschritte | Prozessschritt Pr | Absorber-transitionen

Zustand des Durchlaufens einer sequenzdetektierbaren Gruppe

Erwarteter Zustand nach dem Schalten einer sichtbaren Funktion

Ext. Stellen (Eingänge)

Ext. Stellen (Eingänge)

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\mathbf{11}} = \left(c_{ij}\right) \text{ mit } \left(c_{ij}\right) = \begin{cases} 1\,, & \text{wenn } \exists\, k\text{: } \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in1},jk} = \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{out},ik} \neq 0 \\ 0\,, & \text{sonst} \end{cases},$$

$$= \left(c_{ij}\right) \text{ mit } \left(c_{ij}\right) = \begin{cases} 1, & \text{wenn Ausgangstransition der Gruppe } i \\ & \text{Eingangstransition der Gruppe } j \text{ ist} \\ 0, & \text{sonst} \end{cases} \tag{7.13}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\mathbf{12}} = (c_{ij}) \text{ mit } (c_{ij}) = \begin{cases} 1\,, & \text{wenn } \exists\, k\text{: } \underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{in},jk} = \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{out},ik} \neq 0 \\ 0\,, & \text{sonst} \end{cases},$$

$$= (c_{ij}) \text{ mit } (c_{ij}) = \begin{cases} 1, & \text{wenn Ausgangstransition der Gruppe } i \text{ mit} \\ & \text{Vorbereich der sichtbaren Funktion } j \text{ verknüpft ist} \\ 0, & \text{sonst} \end{cases} \tag{7.14}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\mathbf{21}} = (c_{ij}) \text{ mit } (c_{ij}) = \begin{cases} 1\,, & \text{wenn } \exists\, k\text{: } \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in1},jk} = \underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{out},ik} \neq 0 \\ 0\,, & \text{sonst} \end{cases},$$

$$= (c_{ij}) \text{ mit } (c_{ij}) = \begin{cases} 1, & \text{wenn Nachbereich der sichtbaren Funktion } i \text{ mit} \\ & \text{der Eingangstransition der Gruppe } j \text{ verknüpft ist} \\ 0, & \text{sonst} \end{cases} \tag{7.15}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\mathbf{22}} = (c_{ij}) \text{ mit } (c_{ij}) = \begin{cases} 1\,, & \text{wenn } \exists\, k\text{: } \underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{in},jk} = \underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{out},ik} \neq 0 \\ 0\,, & \text{sonst} \end{cases},$$

$$= (c_{ij}) \text{ mit } (c_{ij}) = \begin{cases} 1, & \text{wenn der Nachbereich der sichtbaren Funktion } i \text{ über eine} \\ & \text{Kommunikationsfunktion mit dem Vorbereich der} \\ & \text{sichtbaren Funktion } j \text{ verknüpft ist} \\ 0, & \text{sonst} \end{cases} \tag{7.16}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\mathbf{13}} = (c_i) \text{ mit } (c_i) = \begin{cases} 1\,, & \text{wenn } \exists\, k\text{: } \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{out},ik} \neq 0 \wedge \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in1},k} = \underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{in},k} = \underline{\mathbf{0}} \\ 0\,, & \text{sonst} \end{cases},$$

$$= (c_i) \text{ mit } (c_i) = \begin{cases} 1, & \text{wenn Prozesschritt } Pr \text{ Ausgangstransition der Gruppe } i \text{ ist,} \\ & \text{nicht Eingangstransition einer Gruppe ist und nicht mit} \\ & \text{dem Vorbereich einer sichtbaren Funktion verknüpft ist} \\ 0, & \text{sonst} \end{cases} \tag{7.17}$$

$$\text{mit } \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in1},k} = (c_i) = \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in1}}(i, j = k)\,, \tag{7.18}$$

$$\text{mit } \underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{in},k} = (c_i) = \underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{in}}(i, j = k)\,, \tag{7.19}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\mathbf{23}} = (c_i) \text{ mit } (c_i) = \begin{cases} 1, & wenn\ \exists\, k\text{: } \underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{out},ik} \neq 0 \wedge \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in1},k} = \underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{in},k} = \underline{\mathbf{0}} \\ 0\,, & sonst \end{cases},$$

$$\text{mit } (c_i) = \begin{cases} 1, & \text{wenn Prozesschritt Pr mit dem Nachbereich der sichtbaren Funk – tion } i \text{ verknüpft ist, nicht Eingangstransition einer Gruppe ist und} \\ & \text{nicht mit dem Vorbereich einer sichtbaren Funktion verknüpft ist} \\ 0, & \text{sonst} \end{cases} \tag{7.20}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\mathbf{31}} = \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in1}}^{\mathrm{T}} \text{ mit } (c_i) = \emptyset, \text{ wenn } \underline{\boldsymbol{C}}_{\mathbf{31},i} = \underline{\mathbf{0}} \wedge \underline{\boldsymbol{C}}_{\mathbf{32},i} = \underline{\mathbf{0}} \wedge \underline{\boldsymbol{C}}_{\mathbf{33},i} = 0 \tag{7.21}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\mathbf{32}} = \underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{in}}^{\mathrm{T}} \text{ mit } (c_i) = \emptyset, \text{ wenn } \underline{\boldsymbol{C}}_{\mathbf{31},i} = \underline{\mathbf{0}} \wedge \underline{\boldsymbol{C}}_{\mathbf{32},i} = \underline{\mathbf{0}} \wedge \underline{\boldsymbol{C}}_{\mathbf{33},i} = 0 \tag{7.22}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\mathbf{33}} = (c_i) \text{ mit } (c_i) = \begin{cases} \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{out},k}^{\mathrm{T}}, & \text{wenn } \exists\, k\colon \sum_{p=1}^{Pr-1} \underline{\underline{\boldsymbol{C}}}_{\mathbf{1},kp} = 0 \\ \underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{out},k}^{\mathrm{T}}, & \text{wenn } \exists\, k\colon \sum_{p=1}^{Pr-1} \underline{\underline{\boldsymbol{C}}}_{\mathbf{2},kp} = 0 \\ \emptyset, & \text{wenn } \underline{\underline{\boldsymbol{C}}}_{\mathbf{31},\mathrm{i}} = \underline{\mathbf{0}} \wedge \underline{\underline{\boldsymbol{C}}}_{\mathbf{32},\mathrm{i}} = \underline{\mathbf{0}} \wedge \underline{\underline{\boldsymbol{C}}}_{\mathbf{33},\mathrm{i}} = 0 \end{cases}$$

$$= (c_i) \text{ mit } (c_i) = \begin{cases} 1, & \text{wenn Kommunikationsfunktion } i \text{ Ausgangstransition einer Gruppe ist, keine Eingangstransition einer Gruppe ist und nicht mit dem Vorbereich einer sichtbaren Funktion verknüpft ist} \\ 1, & \text{wenn Kommunikationsfunktion } i \text{ mit dem Nachbereich einer sichtbaren Funktion verknüpft ist, keine Eingangstransition einer Gruppe ist und nicht mit dem Vorbereich einer sichtbaren Funktion verknüpft ist} \\ 0, & \text{sonst} \end{cases} \tag{7.23}$$

$$\text{mit } \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{out},k} = \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{out}}(i = k, j)\,, \tag{7.24}$$

$$\text{mit } \underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{out},k} = \underline{\underline{\boldsymbol{Ts}}}_{\boldsymbol{out}}(i = k, j)\,, \tag{7.25}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\mathbf{1}} = \left(\underline{\underline{\boldsymbol{C}}}_{\mathbf{11}} \quad \underline{\underline{\boldsymbol{C}}}_{\mathbf{12}}\right), \tag{7.26}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\mathbf{2}} = \left(\underline{\underline{\boldsymbol{C}}}_{\mathbf{21}} \quad \underline{\underline{\boldsymbol{C}}}_{\mathbf{22}}\right), \tag{7.27}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\mathbf{1},kp} = \underline{\underline{\boldsymbol{C}}}_{\mathbf{1}}(i = k,\ j = p), \tag{7.28}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\mathbf{2},kp} = \underline{\underline{\boldsymbol{C}}}_{\mathbf{2}}(i = k,\ j = p), \tag{7.29}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\mathbf{14}} = \left(\underline{\underline{\boldsymbol{C}}}_{\mathbf{14_1}} \quad \cdots \quad \underline{\underline{\boldsymbol{C}}}_{\mathbf{14}_g} \quad \cdots \quad \underline{\underline{\boldsymbol{C}}}_{\mathbf{14}_{Gr}}\right), \tag{7.30}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\mathbf{14_1}} = \left(c_{ij}\right) \text{ mit } (c_i) = \begin{cases} \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in2},1}, & \text{wenn } i = 1 \\ \underline{\mathbf{0}}, & \text{sonst} \end{cases}, \tag{7.31}$$

$$\text{mit } \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in2},1} = \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in2}}(i = 1, j)\,, \tag{7.32}$$

$$\text{mit } (c_j) = \begin{cases} \emptyset, & \text{wenn } \underline{\underline{\boldsymbol{C}}}_{\mathbf{14_1},j} = \underline{\mathbf{0}} \\ (c_j), & \text{sonst} \end{cases}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\mathbf{14}_g} = \left(c_{ij}\right) \text{ mit } (c_i) = \begin{cases} \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in2},g}, & \text{wenn } i = g \\ \underline{\mathbf{0}}, & \text{sonst} \end{cases}, \tag{7.33}$$

$$\text{mit } \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in2},g} = \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in2}}(i = g, j), \tag{7.34}$$

$$\text{mit } (c_j) = \begin{cases} \emptyset, & \text{wenn } \underline{\underline{\boldsymbol{C}}}_{\mathbf{14}_g,j} = \underline{\mathbf{0}} \\ (c_j), & \text{sonst} \end{cases}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{14_{Gr}}} = (c_{ij}) \text{ mit } (c_i) = \begin{cases} \underline{\boldsymbol{Gr}}_{\boldsymbol{in2},Gr}, & \text{wenn } i = Gr \\ \underline{\mathbf{0}}, & \text{sonst} \end{cases}, \tag{7.35}$$

$$\text{mit } \underline{\boldsymbol{Gr}}_{\boldsymbol{in2},Gr} = \underline{\underline{\boldsymbol{Gr}}}_{\boldsymbol{in2}}(i = Gr, j), \tag{7.36}$$

$$\text{mit } (c_j) = \begin{cases} \emptyset, & \text{wenn } \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{14_{Gr},j}} = \underline{\mathbf{0}} \\ (c_j), & \text{sonst} \end{cases}$$

$$\text{mit } \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{34}} = \left(\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{14}}{}^{\mathrm{T}}\, \underline{\underline{\boldsymbol{C}}}_{\boldsymbol{14}}\right) .* \underline{\underline{\boldsymbol{I}}} \tag{7.37}$$

In der Schnittmenge der ersten drei Spalten mit den ersten beiden Zeilen der Blockmatrizen (7.11) und (7.12) wird der vom zu überwachenden Prozessablauf sequenzdetektierbare Bereich abgebildet, der aus zusammenhängenden detektierbaren Gruppen und sichtbaren Funktionen besteht. Die hinterlegten Matrizen $\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{11}}$ bis $\underline{\underline{\boldsymbol{C}}}_{\boldsymbol{23}}$ aus (7.11) und (7.12) ordnen den im sequenzdetektierbaren Bereich vorkommenden Prozesschritten entsprechende Zustände zu, die im Ablauf des überwachten Steuerungsprozesses erwartet werden und die Prozessschritte untereinander verknüpfen.

Die vierte Spalte aus (7.12) repräsentiert über (7.30) und (7.37) Absorbertransitionen zur Durchführung einer Löschoperation im Falle einer Nichterfüllung der Sequenzdetektierbarkeit einer Gruppe. Demnach wird beim Schalten einer Absorbertransition dem Analysemodell die Laufmarke entzogen, falls eine Eingangstransition der Gruppe schaltet, während noch eine Stelle innerhalb der Gruppe markiert ist (Abschnitt 6.3). Aufgrund der dann vorherrschenden Nichterfüllung der Sequenzdetektierbarkeit wird durch das Löschen der Laufmarke der Prozess zur Laufzeitüberwachung abgebrochen.

Die Bereitstellung von Zustandsinformationen aus dem realen Steuerungsprozess erfolgt über externe Eingangsstellen, deren Verknüpfungen in den letzten beiden Zeilen der Blockmatrix (7.12) formuliert sind. Die Verknüpfungen stellen die Verbindungen der externen Stellen mit den Prozessschritten des Analysemodells gemäß (7.21), (7.22) und (7.23) sowie mit den Absorbertransitionen gemäß (7.37) dar.

Die Herausforderung der Laufzeitüberwachung verteilter Systeme besteht darin, aus den durch parallel ablaufende Prozesse erzeugten Daten zusammenhängende Abläufe erkennen zu können. Dazu bedarf es u. a. der detaillierten Kenntnis über zeitliche Interaktionsmuster für das Eintreten von Ereignissen. Die zeitlichen Interaktionsmuster werden beispielsweise in Form von Zeitschranken vorgegeben. Auf Basis der Ergebnisse der Laufzeitermittlung aus Abschnitt 5.3 ließe sich eine verlässlichere Analyse umsetzen, indem nach jedem einzelnen Ereignis in der Steuerung,

einschließlich dem Zyklusübergangsvorgang und dem Funktionsaufruf, eine Laufzeitprüfung durchgeführt werden kann. An dieser Stelle seien dennoch folgende Problemstellungen genannt:

- Sämtliche der Laufzeitüberwachung heranzuziehenden Ereignisse aus der Steuerung müssen im Analysemodell über Prozessschritte abbildbar sein. Die Aktivierung der Prozessschritte ist somit über sämtliche Zustände aus dem Steuerungsprozess zu realisieren. Mit der realisierten Petrinetzumgebung aus Abschnitt 7.1 lassen sich jedoch nur Zustände aus dem Steuerungsprozess auslesen, die einen Kommunikationsprozess nach sich ziehen.

- Unter Berücksichtigung sämtlicher Ereignisse für die Laufzeitüberwachung muss das betrachtete Gesamtsystem weiterhin eine chronologische Totalordnung aufweisen, sofern eindeutig auf eine Kausalbeziehung zwischen den eintretenden Ereignissen geschlossen werden soll.

Anwendungsbeispiel

In Abbildung 7.7 ist ein Analysemodell zur Laufzeitprüfung für den Fall dargestellt, dass alle Ereignisse innerhalb des in Abschnitt 5.3 eingeführten Steuerungsprozesses detektierbar sind. Damit wären äquivalent zu allen Ereignissen aus der Steuerung die modellierten Prozesschritte innerhalb des Analysemodells aus Abbildung 7.7 aktivierbar. Eine Modellerstellung auf Basis der Analyse der Sequenzdetektierbarkeit ist in diesem Fall nicht notwendig, da die Eigenschaft der Ereignisdetektierbarkeit für den Realprozess vorliegt. Entsprechend bildet das Modell aus Abbildung 7.7 alle Prozesschritte in Form von sichtbaren Transitionen ab. Die Berücksichtigung sequenzdetektierbarer Gruppen und die Modellierung von Absorbertransitionen sind demnach also nicht erforderlich. Für den abgebildeten Steuerungsprozess gelten die minimale erwartete Laufzeit $t_{seq,min} = 5{,}5\ \mathrm{s}$ und die maximal erwartete Laufzeit $t_{seq,max}$ = 15 s gemäß den Ergebnissen aus Tabelle 12.

Unter Anwendung der vorgestellten Online-Analyseumgebung (Abschnitt 7.1), die ausschließlich auf einer Detektierbarkeit versandter Bustelegramme beruht, ist eine Laufzeitüberwachung mittels des in Abbildung 7.7 dargestellten Modells jedoch nicht möglich, da nur eine Teilmenge an Zuständen aus dem kausalen Prozessablaufes als Ereignis das Absenden eines Bustelegrammes nach sich zieht, sodass der Steuerungsprozess nicht vollständig ereignisdetektierbar ist. Folglich sind die im Analysemodell aus Abbildung 7.7 grau dargestellten externen Stellen nicht aus der Telegrammübergabe-Ebene heraus markierbar.

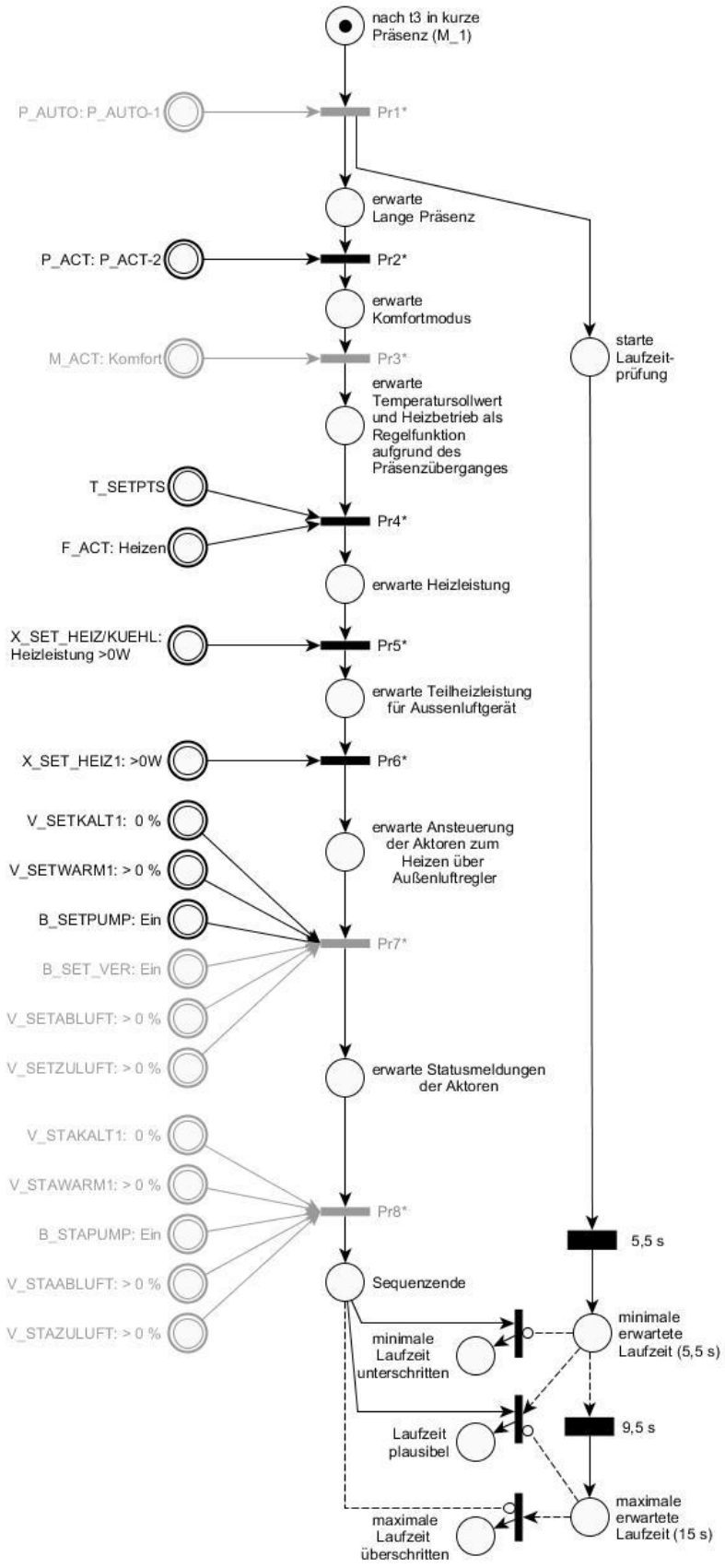


Abbildung 7.7: Soll-Modell zur Laufzeitüberwachung eines exemplarischen Prozessablaufes

Für eine anwendbare Analyse ist der Prozessablauf daher auf das abstrahierte Analysemodell aus Abbildung 7.8 zu beschränken. Demnach wird für das Modell zur Laufzeitprüfung lediglich die Auswertung der zusammenhängenden Gruppen *Gr2* und *Gr3* sowie der sichtbaren Transition *Fkt21* berücksichtigt, d. h. der sequenzdetektierbare Bereich des kausalen Ablaufes. Die Laufzeitüberwachung beginnt mit der Detektion des Zustandes der langen Präsenz P_ACT:P_ACT-2 und endet nach dem Empfang der Teilleistung X_SET_HEIZ1 für das Außenluftgerät.

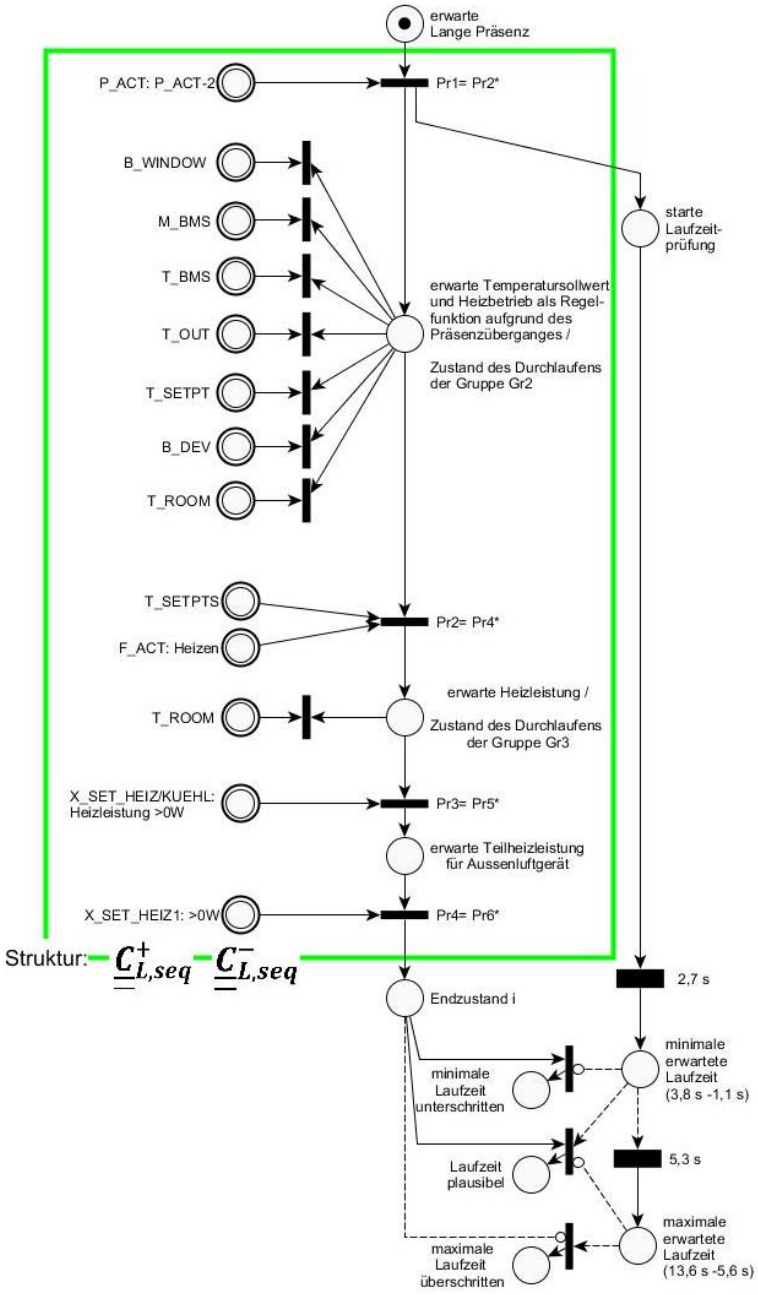


Abbildung 7.8: Anwendbares Modell zur Laufzeitüberwachung eines exemplarischen Prozessablaufes

Das in Abbildung 7.8 hellgrün umrandete Modellkonstrukt beruht auf der Strukturbeschreibung nach (7.11) und (7.12) und lässt sich wie folgt beschreiben:

$$\underline{\underline{\boldsymbol{C}}}^{+}_{\boldsymbol{L,seq}} = \begin{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} & \underline{\mathbf{0}} & \underline{\mathbf{0}} & \underline{\mathbf{0}} \\ \underline{\mathbf{0}} & 1 & \underline{\mathbf{0}} & \underline{\mathbf{0}} \\ \underline{\mathbf{0}} & \underline{\mathbf{0}} & \underline{\mathbf{0}} & \underline{\mathbf{0}} \\ \underline{\mathbf{0}} & \underline{\mathbf{0}} & \underline{\mathbf{0}} & \underline{\mathbf{0}} \end{pmatrix}$$

4 − 1 Prozessschritte | Prozessschritt 4 | Absorber-transitionen

Zustand des Durchlaufens einer sequenzdetektierbaren Gruppe

Erwarteter Zustand nach dem Schalten einer sichtbaren Funktion

Ext. Stellen (Eingänge)

und

$$\underline{\underline{\boldsymbol{C}}}^{-}_{\boldsymbol{L,seq}} = \begin{pmatrix} \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} & \begin{matrix} 0 \\ 0 \\ 1 \end{matrix} & \begin{matrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{matrix} \\ \underline{\mathbf{0}} & & \\ \begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{matrix} & \begin{matrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{matrix} & \\ \underline{\mathbf{0}} & & \\ \underline{\mathbf{0}} & \underline{\mathbf{0}} & \underline{\mathbf{0}} \end{pmatrix}$$

$$\begin{matrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{matrix}$$

4 − 1 Prozessschritte | Prozessschritt 4 | Absorber-transitionen

Zustand des Durchlaufens einer sequenzdetektierbaren Gruppe

Erwarteter Zustand nach dem Schalten einer sichtbaren Funktion

Ext. Stellen (Eingänge) .

Die Matrizen beinhalten u.a. die Beschreibung der Verknüpfung von vier Prozesschritten *Pr1* bis *Pr4* innerhalb des Analysemodells, die das Ereignis des Starts der Laufzeitprüfung (äquivalent der Eingangstransition der Gruppe *Gr2*) sowie das erfolgreiche Durchlaufen der Gruppen *Gr2* und *Gr3* bzw. der sichtbaren Funktion Fkt21 abbilden. Die Aktivierung der Prozesschritte erfolgt in Abhängigkeit der Belegung der externen Eingangsstellen P_ACT:P_ACT-2, T_SETPTS, F_ACT:Heizen, X_SET_HEIZ/KUEHL und X_SET_HEIZ1. Deren Belegung erfolgt auf Basis der Ausführung und Auswertung der Kommunikationsprozesse, die direkter Bestandteil des kausalen Prozessablaufes sind (Abschnitt 6.3). Beispielsweise erfolgt der Prozessschritt

Pr2, der das Ereignis des erfolgreichen Durchlaufens der Gruppe *Gr2* abbildet, sobald entsprechend der Belegung der externen Stellen T_SETPTS und F_ACT:Heizen die vom Realssystem auf dem Bus abgelegten Telegramme (Tel13 und Tel14 aus Abbildung 6.4) von der Analyseumgebung empfangen und ausgewertet sind. Das Modell aus Abbildung 7.8 besteht weiterhin aus acht Absorbertransitionen, die bei Nichteinhaltung der Sequenzdetektierbarkeit einer Gruppe (ungewünschtes Schalten einer Eingangstransition während des Durchlaufens seiner Gruppe) die Laufmarke aus dem Analysemodell abzieht und damit die Laufzeitprüfung abbricht (Abschnitt 6.3). Befindet sich das Analysemodell im Zustand des Durchlaufens der Gruppe *Gr2* würde eine solche Löschoperation durchgeführt werden, sobald eine der Eingangstransitionen der Gruppe schaltet, die den externen Stellen B_WINDOW, M_BMS, T_BMS, T_OUT, T_SETPT, B_DEV oder T_ROOM entspricht. Im Falle der Markierung des Zustandes *erwarte Heizleistung* (Durchlaufen der Gruppe *Gr3*) würde die Laufmarke entzogen werden sofern die externe Stelle T_ROOM belegt wird. Im Vergleich zu den externen Stellen, deren Belegung die Aktivierung der Prozesschritte *Pr1* bis *Pr4* herbeiführen, sind die an die Absorbertransitionen geknüpften externen Stellen nicht direkter Bestandteil des zu untersuchenden kausalen Prozessablaufes (Abschnitt 6.3).

Bei der Parametrisierung des Modells aus Abbildung 7.8 ist zu berücksichtigen, dass sich im Vergleich zu dem Modell aus Abbildung 7.7 veränderte Laufzeitschranken ergeben, da lediglich der sequenzdetektierbare Bereich des kausalen Ablaufes abgebildet und hinsichtlich der Laufzeit überwacht werden kann. Die geltenden Laufzeitschranken können aus den in Abschnitt 5.3 vorgestellten Ergebnissen zur Laufzeitermittlung bestimmt werden:

$$t_{seq,min} = t_{Kom4 \rightarrow \mathrm{Kom22},min}$$

$$= t_{Fkt3 \rightarrow \mathrm{Kom22},min} - t_{Fkt3 \rightarrow Kom4,min} = 3{,}8\ \mathrm{s} - 1{,}1\ \mathrm{s} = 2{,}7\ \mathrm{s} \qquad (7.38)$$

und

$$t_{seq,max} = t_{Kom4 \rightarrow \mathrm{Kom22},max}$$

$$= t_{Fkt3 \rightarrow \mathrm{Kom22},max} - t_{Fkt3 \rightarrow Kom4,max} = 13{,}6\ \mathrm{s} - 5{,}6\ \mathrm{s} = 8\ \mathrm{s}\,. \qquad (7.39)$$

Die minimale und maximale Laufzeit beschänkt sich somit je auf das Intervall zwischen der Kommunikationsfunktion Kom4 (entsprechend dem Datum P_ACT) und der Kommunikationsfunktion Kom22 (entsprechend dem Datum X_SET_HEIZ1), konform mit dem Beginn und dem Ende des sequenzdetektierbaren Bereiches des kausalen Prozessablaufes.

8 Zusammenfassung und Ausblick

In Rahmen der vorliegenden Arbeit wurde ein systematisches Vorgehen für eine vereinheitlichte Beschreibung, Modellierung und Analyse eines verteilten Automatisierungssystems vorgestellt und zur Ermittlung von unbekannten Systemeigenschaften am Beispiel eines Raumautomationssystems angewendet. In Anlehnung an die festgelegte Zielsetzung aus Abschnitt 1.4 werden im folgenden Abschnitt 8.1 die erlangten Ergebnisse zusammengefasst. Ansatzpunkte für weiterführende Arbeiten werden im abschließenden Abschnitt 8.2 dargelegt.

8.1 Zusammenfassung

Ein Konzept für eine effektive und phasenübergreifende Analyse eines verteilten Automatisierungssystems wurde erarbeitet, welches die Nutzung weniger ausgewählter Betriebsmittel (BMW), deren Verzahnung untereinander sowie das Einbeziehen des Realsystems vorsieht. Das Analysevorgehen basiert auf vorhandenen Methoden zur klassischen Systemanalyse, zur zielorientierten (De-)Komposition eines Systems und zur Begriffsformalisierung. Als Beschreibungsmittel sind Klassendiagramme, Petrinetze und Formulierungen mittels Differentialgleichungen als relevant beurteilt worden. Die Auswahl verwendeter Werkzeuge wurde im Wesentlichen auf Grundlage von vorhandenen und geeigneten Übergabestellen getroffen, die die Kopplung der Werkzeuge untereinander und die Kopplung der Werkzeuge an ein Realsystem bieten. Weiterhin wurden bei der Auswahl der Werkzeuge Effizienzeigenschaften berücksichtigt um insbesondere bei der Ausführung hochskalierter Modelle eine Reduzierung der Simulationszeiten zu erreichen.

Die Basis des Analysekonzeptes stellen wissensbeschreibende Systemmodelle dar, welche in Bezug auf eine konkrete Problemstellung eindeutige und geeignete Abbildungen von Begrifflichkeiten und Informationen bereitstellen. Die wissensbeschreibende Systemmodellierung umfasst eine systematische Erhebung und Quantifizierung von Eigenschaften und Anforderungen auf verschiedenen Abstraktionsniveaus, beginnend von der abstraktesten Sicht auf den allgemeinen Systembegriff bis hin zur gerätetechnischen Sicht auf ein konkretes Bauelement und seine Kennwerte. Exemplarisch wurde das Wissen über das Automatisierungssystem SmallCAN präzise in Form solcher Systemmodelle beschrieben, so dass sich aus den dort hinterlegten Informationen ausführbare Analysemodelle gewinnen ließen.

Für eine automatisierte Erstellung ausführbarer Analysemodelle zu konkreten Problemstellungen wurden folgende Modellstrukturen mathematisch beschrieben:

- Deskriptives Modell des Bitübertragungssystems als Voraussetzung für die Analyse des Signalverhaltens auf Datenleitungen
- Simulationsmodell des Kommunikationssystems zur Analyse des Verzögerungsverhaltens aufgrund erhöhter Busauslastung
- Simulationsmodell zur Analyse des Verzögerungsverhaltens aufgrund von Zyklusübergängen zwischen Programmkomponenten
- Simulationsmodell eines Steuerungsprozesses zur Ermittlung von Laufzeiten
- Strukturmodell für die Analyse der Sequenzdetektierbarkeit eines Prozessablaufes als Kriterium der Beobachtbarkeit
- Implementierungsmodell eines ausführbaren Testfalls für eine Funktionsprüfung
- Monitormodell zur Überwachung der Laufzeit eines kausalen Steuerungsprozesses.

Mit Hilfe eines Vergleiches zwischen Mess- und Simulationsergebnissen wurden die Simulationsmodelle validiert, die für die Bewertung der Signalqualität und für die Analyse von Verzögerungs- und Laufzeiten herangezogen wurden.
Mit der Ausführung hochskalierter Simulationsmodelle ließen sich bisher unbekannte Systemeigenschaften und Kennwerte komplexer Instanzen des betrachteten Automatisierungssystem SmallCAN ermitteln bzw. die Einhaltung erhobener Anforderungen beurteilen. Für die genannten (nicht-)funktionalen Eigenschaften konnten so folgenden Ergebnisse erzielt werden.

- Dem betrachteten Kommunikationssystem kann auch unter maximaler räumlicher Ausdehnung und maximaler Anzahl an Kommunikationsteilnehmern eine ausreichende Signalqualität zugesprochen werden.
- Auf Basis von Untersuchungen hinsichtlich der elektromagnetischen Verträglichkeit wurden geeignete Maßnahmen gefunden, die die normativen Anforderungen für den Gebrauch des Systems SmallCAN in Wohnbereichen, Geschäfts- und Gewerbebereichen sowie Kleinbetrieben erfüllen.

- Für das betrachtete Kommunikationssystem wurde das Verzögerungsverhalten bei erhöhter Busauslastung analysiert. Weiterhin wurde das Zyklusübergangsverhalten eines verteilten Programmsystems untersucht. Es konnten jeweils zeitbezogene Kennwerte wie beispielsweise maximal zu erwartende Verzögerungszeiten ermittelt werden.

- Durch eine realitätsnahe Simulation eines Steuerungsprozesses, der durch das folgegesteuerte Ausführen verteilter Verarbeitungsfunktionen entsteht, konnten unter Berücksichtigung sämtlicher Verzögerungszeiten restriktive Kennwerte für die Laufzeiten in Form von Zeitschranken ermittelt werden.

Mit Hilfe von Strukturmodellen wurden die Eigenschaften der Beobachtbarkeit und der Steuerbarkeit am Beispiel der Testbarkeit einer verarbeitenden Funktion sowie am Beispiel der Laufzeitüberwachbarkeit einer Funktionskette analysiert. Durch die strukturellen Untersuchungen konnte vorab die Umsetzbarkeit einer Online-Analyse, z. B. einer Testfallausführung bzw. einer Laufzeitüberwachung, geprüft werden.

Zur Ausführung eines Testfalles in der Phase der Inbetriebnahme und zur Durchführung einer Laufzeitüberwachung in der Phase des operativen Betriebes wurde eine Petrinetz-basierte Umgebung für die Online-Analyse entwickelt, die das Automatisierungssystem SmallCAN an das jeweilige Online-Analysemodell (Implementierungsmodell eines ausführbaren Testfalls bzw. Monitormodell zur Überwachung der Laufzeit) koppelt.

Zusammenfassend lässt sich festhalten, dass die Anwendung eines strukturierten Analysevorgehens auf Grundlage eines wissensbeschreibenden begriffs-formalisierenden Systemmodells und einer vereinheitlichten automatisierten Ausführungs-Umgebung auf Basis mathematisch beschreibbarer und automatisiert skalierbarer Analysemodellen es zum einen ermöglicht das Systemverständnis komplexer Automatisierungssysteme tiefgreifender zu durchdringen und zum anderen den steigenden Anforderungen an der Analyse solcher Systeme Gerecht zu werden.

8.2 Ausblick

Die folgenden Vorschläge für weiterführende Arbeiten betreffen das dem Analysekonzept zugrundeliegende Beschreibungsmittel der Petrinetze sowie die allgemeine Anwendbarkeit der wissensrepräsentierenden Systemmodelle:

- Nahezu alle ausführbaren Analysemodelle wurden mit Petrinetzen erstellt. Lediglich die Untersuchungen der Signalqualität erfolgten über SPICE-basierte Beschreibungsmittel. Um der Konzeptanforderung nach der Verwendung

möglichst weniger Beschreibungsmittel gerecht zu werden, ist es zielführend für die Analyse von Systemen mit wertkontinuierlichen Zuständen ebenfalls Petrinetze einzusetzen (kontinuierliche oder hybride Petrinetze).

Im Falle der Online-Analyse sind Petrinetze ausschließlich an den Beispielen einer Testfallausführung und einer Laufzeitüberwachung erläutert worden. Es bieten sich daher weiterführende Arbeiten an, die die Nutzung Petrinetz-basierter Modelle für eine Online-Analyse eines verteilten Automatisierungssystems behandeln.

- Die wissensbeschreibenden Systemmodelle sind in dieser Arbeit beispielhaft für das konkrete Automatisierungssystem SmallCAN behandelt und manuell entwickelt worden. Für eine allgemeingültige Anwendbarkeit der Systemmodelle ist es zielführend, diese mittels grundlegenden Systemwissen über verteilte Automatisierungssysteme abstrakter zu formulieren und werkzeugbasiert, beispielsweise in *iglos* [Stein et al. 2010: 18], zu erstellen.

  Für den Informationsaustausch zwischen den wissensbeschreibenden Systemmodellen und den ausführbaren Analysemodellen empfiehlt sich ein einheitliches Datenaustauschformat wie AutomationML [Diekhake et al. 2016].

9 Literaturverzeichnis

[BSI TR-03109-1:2013] Bundesamt für Sicherheit in der Informationstechnik: *Anforderungen an die Interoperabilität der Kommunikationseinheit eines intelligenten Messsystems (BSI TR-03109-1)*, 1, 2013.

[DIN EN 15232:2012] DIN EN 15232: *Energieeffizienz von Gebäuden - Einfluss von Gebäudeautomation und Gebäudemanagement*, 2012.

[DIN EN 55016-2-3:2006] DIN EN 55016-2-3: *Verfahren zur Messung der hochfrequenten Störaussendung (Funkstörungen) und Störfestigkeit - Messung der gestrahlten Störaussendung*, 2006.

[DIN IEC 60050-351:2014] DIN IEC 60050-351: *Internationales Elektrotechnisches Wörterbuch*, 2014.

[DIN EN 61000-4-2:2005] DIN EN 61000-4-2: *Prüf- und Messverfahren - Prüfung der Störfestigkeit gegen die Entladung statischer Elektrizität*, 2005.

[DIN EN 61000-4-3:2002] DIN EN 61000-4-3: *Prüf- und Messverfahren - Prüfung der Störfestigkeit gegen hochfrequente elektromagnetische Felder*, 2002.

[DIN EN 61000-4-4:2004] DIN EN 61000-4-4: *Prüf- und Messverfahren - Prüfung der Störfestigkeit gegen schnelle transiente elektrische Störgrößen/Burst*, 2004.

[DIN EN 61000-4-5:1995] DIN EN 61000-4-5: *Prüf- und Messverfahren - Prüfung der Störfestigkeit gegen Stoßspannungen*, 1995.

[DIN EN 61000-4-6:2007] DIN EN 61000-4-6: *Prüf- und Messverfahren - Störfestigkeit gegen leitungsgeführte Störgrößen, induziert durch hochfrequente Felder*, 2007.

[DIN EN 61000-6-2:2005] DIN EN 61000-6-2: *Elektromagnetische Verträglichkeit (EMV) - Teil 6-2: Fachgrundnormen - Störfestigkeit für Industriebereiche*, 2005.

[DIN EN 61000-6-3:2011] DIN EN 61000-6-3: *Elektromagnetische Verträglichkeit (EMV) - Teil 6-3: Fachgrundnormen - Störaussendung für Wohnbereich, Geschäfts- und Gewerbebereiche sowie Kleinbetriebe*, 2011.

[DIN EN 61499-1:2014] DIN EN 61499-1: *Funktionsbausteine für industrielle Leitsysteme - Teil 1: Architektur*, 2014.

[DIN 69901-1:2009] DIN 69901-1: *Projektmanagement - Projektmanagementsysteme - Teil 1: Grundlagen*, 2009.

[EU 2010/31] EU 2010/31: *Richtlinie 2010/31/EU des Europäischen Parlaments und des Rates vom 19. Mai 2010 über die Gesamtenergieeffizienz von Gebäuden Absatz*, 2010.

[EEG 2014]: *Gesetz für den Vorrang Erneuerbarer Energien (Erneuerbare - Energien - Gesetz - EEG) (EEG)*, 2014.

[EnEV 2014]: *Verordnung über energiesparenden Wärmeschutz und energiesparende Anlagentechnik bei Gebäuden (Energieeinsparverordnung - EnEV) (EnEV)*, 2014.

[VDI 3813:2011] VDI Verein Deutscher Ingenieure e.V.: *Gebäudeautomation (GA) - Raumautomation (VDI 3813)*, 2011.

[VDI 3814:2009] VDI Verein Deutscher Ingenieure e.V.: *Gebäudeautomation (GA) (VDI 3814)*, 2009.

[Abel 1990] Abel, Dirk: *Petri-Netze für Ingenieure*, Springer, Berlin, 1990.

[acatech 2013] acatech: *Umsetzungsempfehlungen für das Zukunftsprojekt Industrie 4.0. Abschlussbericht des Arbeitskreises Industrie 4.0*, 2013.

[Aguirre-Salas/Santoyo-Sanchez 2009] Aguirre-Salas, Luis und Santoyo-Sanchez, Alejandra: *Sequence-detectability analysis of interpreted Petri nets under partial state observations*. In: IEEE (Hrsg.): Factory Automation. ETFA, S. 1–7, 2009.

[AimoneMarsan et. al. 1986] Aimone Marsan, Marco; Balbo, Gianfranco; Conte, Gianni.: *Performance Models of Multiprocessor Systems*. In: MIT Press, London, 1986.

[Alt 2012] Alt, Oliver: *Modellbasierte Systementwicklung mit SysML* : Hanser, 2012.

[Antoniou et al. 2002] Antoniou, I.; Ivanov, V.V; Ivanov, Valery V. und Zrelov, P.V: *On the log-normal distribution of network traffic*, In: Physica D: Nonlinear Phenomena, 167 (2002) Heft 1-2, S. 72–85, 2002.

[Bachmann 2013] Bachmann, Thomas: *Dynamisches Prüfen von Softwarekomponenten eines verteilten Automatisierungssystems basierend auf Petri-Netzen* : Masterarbeit, Technische Universität Braunschweig, Braunschweig, 2013.

[Baumgarten 1997] Baumgarten, Bernd: *Petri-Netze: Grundlagen und Anwendungen* : Spektrum Akademischer Verlag, 1997.

[Berger et al. 2012] Berger, Ulrich; Schöttke, Dirk und Kämpfe, Thomas: *Das Zeitverhalten verteilter Anlagen*. In: Deutscher Industrieverlag (Hrsg.): atp-Automatisierungstechnische Praxis, S. 36–43, 2012.

[Bidjan-Irani 1989] Bidjan-Irani, Mehrdad: *Qualität und Testbarkeit hochintegrierter Schaltungen. Qualitätssicherung durch regelbasierte Systeme*. Nr. 215, Springer, Berlin, New York, 1989.

[Binder 1994] Binder, Robert V.: *Design for testability in object-oriented systems*, In: Communications of the ACM, 37 (1994) Heft 9, S. 87–101, 1994.

[Bredebusch 1994] Bredebusch, A.: *A Petri-net representation of the qualitative behaviour of a dynamical continuous-time system*. In: Second International Conference on `Intelligent Systems Engineering', S. 223–228, 1994.

[Brackmann/Varchmin 1996] Brackmann, Ludwig; Varchmin, J.-Uwe: *Das intelligente Haus wird Wirklichkeit*, In: Mitteilungen der Technischen Universität Carolo-Wilhelmina zu Braunschweig, Jahrgang XXXI, Heft 1, S. 53-68, 1996.

[Brackmann/Varchmin 1997] Brackmann, Ludwig; Varchmin, J.-Uwe: *Häusliche Automation mit European Home Systems*, In: Spektrum der Wissenschaft 11/1997, S. 102, 1997.

[Bredebusch 1994] Bredebusch, A.: *A Petri-net representation of the qualitative behaviour of a dynamical continuous-time system*. In: Second International Conference on `Intelligent Systems Engineering', S. 223–228, 1994.

[Bussenius 2014] Bussenius, Steffen: *Markanalyse für den Markteintritt des Produktes SmallCAN anhand eines Businessplans*: Masterarbeit, Ostfalia Hochschule für angewandte Wissenschaften, Wolfenbüttel, 2014.

[Cassandras/Lafortune 2008] Cassandras, Christos G. und Lafortune, Stéphane: *Introduction to discrete event systems*. 2nd, Springer, New York, 2008.

[Chu 2014] Chu, Yiwen: *Methodische Analyse der unteren Schichten des ISO/OSI Referenzmodells am Beispiel des Feldbussystems SmallCAN* : Masterarbeit, Technische Universität Braunschweig, Braunschweig, 2014.

[Coulouris et al. 2002] Coulouris, George; Dollimore, Jean und Kindberg, Tim: *Verteilte Systeme. Konzepte und Design*. 3., überarb. Aufl., Pearson Education Deutschland, München, 2002.

[Dadji et al. 2010] Dadji, Yannick; Michalik, Harald; Kohn, Nnamdi; Steiner, Jans; Beckmann, Guido; Möglich, Tobias und Varchmin, Jörn-Uwe: *A Communication Architecture for Distributed Real-Time Robot Control*. In: Springer (Hrsg.): Robotic Systems for Handling and Assembly, S. 213–231, 2010.

[David/Alla 2001] David, René und Alla, Hassane: *On Hybrid Petri Nets*, In: Discrete Event Dynamic Systems, 11 (2001) Heft 1/2, S. 9–40, 2001.

[Déqué et al. 2000] Déqué, F.; Ollivier, F. und Poblador, A.: *Grey boxes used to represent buildings with a minimum number of geometric and thermal parameters*, In: Energy and Buildings, 31 (2000) Heft 1, S. 29–35, 2000.

[Diekhake 2011] Diekhake, Patrick: *Buslast- und Schaltungssimulation zur Validierung des optimierte Feldbussystems SmallCAN bei maximaler Auslastung*. In: Kommunikation in der Automation (KommA), 2011.

[Diekhake et al. 2011] Diekhake, Patrick; Fähnrich, Elmar; Schnieder, Eckehard und Becker, Uwe: *SmallCAN: Integrierte Gebäudeautomatisierung durch einheitliches low-Power, low-Cost, OpenSource Feldbussystem*. In: VDI (Hrsg.): Automation 2011, S. 105–109, Düsseldorf, 2011.

[Diekhake 2013] Diekhake, Patrick: *Physical Layer Simulation of Large Distributed Automation Systems in SPICE*. In: IARIA (Hrsg.): International Conference on Advances in System Simulation (SIMUL), S. 165–169, 2013.

[Diekhake et al. 2014] Diekhake, Patrick; Schnieder, Eckehard und Becker, Uwe: *Demonstrationsanlage einer integrierten Gebäudeautomatisierung mit low-Power, low-Cost Ansatz und flexiblem Gerätespektrum und flexibler Konfiguration (DIGAFLEX)*. In: EnOB (Hrsg.): Energieinnovationen in Neubau und Sanierung. Neues aus der Forschung für mehr Energieeffizienz, Raumkomfort, Wirtschaftlichkeit und Nachhaltigkeit, S. 31, 2014.

[Diekhake 2014] Diekhake, Patrick: *Untersuchung von Smart Grid Anwendungen am Beispiel einer Klima- und Temperaturregelung unter Verwendung eines einheitlich automatisierten Demonstrators zur Validierung eines aktiven Verteilnetzes*. In: Branchentreff der Mess- und Automatisierungstechnik (AUTOMATION), 2014.

[Diekhake/Schnieder 2013] Diekhake, Patrick und Schnieder, Eckehard: *Online monitoring of a distributed building automation system to verify large sequences of bus messages by causal Petri net models*. In: IECON - 39th Annual Conference of the IEEE Industrial Electronics Society, S. 3651–3655, 2013.

[Diekhake/Schnieder 2015] Diekhake, Patrick und Schnieder, Eckehard: *Strukturierte Modellierung, Simulation und Überwachung verteilter Automatisierungssysteme*, In: at - Automatisierungstechnik, 63 (2015) Heft 2, 2015.

[Diekhake et al. 2016] Diekhake, Patrick; Diaz, Damian; Becker, Uwe; Günther, Michelle; Scholz, André; Puntel Schmidt, Philipp; Fay, Alexander: *Bewertung verteilter Gebäudeautomatisierungssysteme auf Basis ihrer Beschreibung mittels AutomationML und ihrer Simulation unter Anwendung automatisiert erzeugter Petrinetze.* In: Entwurf komplexer Automatisierungssysteme (EKA), 2016.

[Ding et al. 2009] Ding, Liangxue; Shen, Yi und Zhou, Yang: *Real time performance analysis and evaluation of CAN bus with an extended Petri Net model*. In: IEEE (Hrsg.): Intrumentation and Measurement Technology Conference. I2MTC, S. 1081–1084, 2009.

[Donath et al. 1995] Donath, Ulrich; Haufe, Jürgen und Schwarz, Pauline: *Mehr-Ebenen-Simulation automatisierungstechnische Prozesse und Steuerungen*. In: Entwurf komplexer Automatisierungssysteme (EKA), S. 543–553, 1995.

[Dongbo et al. 2008] Dongbo, Pan; Feng, Liu; Xuelian, Zhou und Tao, Li: *Functional safety in building automation and control systems*. In: 2008 3rd IEEE Conference on Industrial Electronics and Applications (ICIEA), S. 467–470, 2008.

[Eichelberg 2010] Eichelberg, Marco: *Interoperabilität von AAL-Systemkomponenten*, VDE-Verl, Berlin, Offenbach, 2010.

[Eickmeyer et al. 2012] Eickmeyer, Philipp; Diekhake, Patrick; Goydke, Hans und Huckemann Volker: *future:workspace - Arbeitswelten des 21.Jahrhunderts auf dem Prüfstand*. In: Bauverlag (Hrsg.): TAB Technik am Bau, S. 28–34, 2012.

[Fay et al. 2008] Fay, Alexander; Jerenz, Stefan; Seitz, Nando: *Dezentrale Steuerung von Transportsystemen in Analogiezum Routing in Datennetzen*. In: at - Automatisierungstechnik, 56 (2008) Heft 6, S. 284–295, 2008.

[Föllinger et al. 1994] Föllinger, Otto; Dörrscheidt, Frank und Klittich, Manfred: *Regelungstechnik. Einführung in die Methoden und ihre Anwendung*. 8., überarb. Aufl. Aufl., Hüthig, Heidelberg, 1994.

[Fountain et al. 1996] Fountain, Marc; Brager, Gail und Dear, Richard de: *Expectations of indoor climate control*, In: Energy and Buildings, 24 (1996) Heft 3, S. 179–182, 1996.

[Frey/Litz 2000] Frey, Georg. und Litz, Lothar: *Steuerungsentwurf mit Petrinetzen*, In: Automatisierung + Datentechnik, (2000) Heft 45, S. 61–62, 2000.

[Georgieff 2008] Georgieff, Peter: *Ambient Assisted Living. Marktpotenziale IT-unterstützter Pflege für ein selbstbestimmtes Altern*, Stuttgart, 2008.

[Gevatter/Grünhaupt 2004] Gevatter, Hans-Jürgen und Grünhaupt, Ulrich (Hrsg.): *Mess- und Automatisierungstechnik im Automobil*. 2., vollst. überarb. Aufl, Springer, Berlin, 2004.

[Giua 1997] Giua, Alessandro: *Petri net state estimators based on event observation*. In: IEEE (Hrsg.): 36th Conference on Decision and Control. CDC, S. 4086–4091, 1997.

[Giua/Seatzu 2014] Giua, Alessandro und Seatzu, Carla: *Petrinetze und die Steuerung Ereignisdiskreter Systeme*, In: Informatik-Spektrum, (2014) Heft 37, S. 199–210, 2014.

[Girault/Valk 2003] Girault, Claude; Valk, Rudiger: *Petri Nets for System Engineering: A Guide to Modeling, Verification, and Applications*, Springer, 2003.

[Gomes et al. 2007] Gomes, Luis; Costa, Aniko; Barros, Joao Paulo; Pais, Rui; Rodrigues, Tiago und Ferreira, Richard: *Petri Net based Building Automation and Monitoring System*. In: 5th IEEE International Conference on Industrial Informatics, S. 57–62, 2007.

[Graditi et al. 2013] Graditi, Giorgio; Atrigna, Mauro; Piccolo, Antonio und Siano, Pierluigi: *Energy management system for smart homes: Testing methodology and test-case generation*. In: International Conference on Clean Electrical Power (ICCEP), S. 766–771, 2013.

[Granzer et al. 2010a] Granzer, Wolfgang; Reinisch, Christian und Kastner, Wolfgang: *Future challenges for building automation: Wireless and security*. In: IEEE International Symposium on Industrial Electronics (ISIE 2010), S. 4415–4467, 2010.

[Granzer et al. 2010b] Granzer, Wolfgang; Praus, Fritz und Kastner, Wolfgang: *Security in Building Automation Systems*, In: IEEE Transactions on Industrial Electronics, 57 (2010) Heft 11, S. 3622–3630, 2010.

[Granzer/Kastner 2010] Granzer, Wolfgang und Kastner, Wolfgang: *Security Analysis of Open Building Automation Systems*. In: Computer Safety, Reliability, and Security, S. 303–316. Springer, Berlin Heidelberg, 2010.

[Greifeneder et al. 2007] Greifeneder, Jürgen; Liu, Liu und Frey, Georg: *Methoden zur Antwortzeitanalyse in vernetzten Automatisierungssystemen*. In: SPS/IPC/DRIVES, S. 517–525, Nürnberg, 2007.

[Guezgouz et al. 2010] Guezgouz, D.; Raingeaud, Y. und Lebunetel, J-C.: *SPICE model for the PLC propagation channel in the high frequency range*. In: ISPLC, S. 1–6, 2010.

[Gunther et al. 2010] Gunther, Harald; Frei, Stephan und Wenzel, Thomas: *Simulation methods for signal integrity of automotive bus systems*. In: IEEE (Hrsg.): Asia-Pacific International Symposium on Electromagnetic Compatibility. APEMC, S. 512–515, 2010.

[Hanisch 1992] Hanisch, Hans-Michael: *Petri-Netze in der Verfahrenstechnik. Modellierung und Steuerung verfahrenstechnischer Systeme*, Oldenbourg, München [u.a.], 1992.

[Helbig 2008] Helbig, Hermann: *Wissensverarbeitung und die Semantik der natürlichen Sprache. Wissensrepräsentation mit MultiNet*. [Online-Ausg. der] 2., überarb. [gedr.] Aufl., Springer, Berlin, Heidelberg, 2008.

[Hesse et al. 1994] Hesse, Wolfgang; Barkow, Gert; Braun, Hubert. von; Kittlaus, Hans-Bernd und Scheschonk, Gert: *Terminologie der Softwaretechnik. Ein Begriffssystem für die Analyse und Modellierung von Anwendungssystemen,Teil 1:Begriffssystematik und Grundbegriffe*, In: Informatik-Spektrum (1994) Heft 17, S. 39–47, 1994.

[Höme/Dietrich 2012] Höme, Stephan Höme und Dietrich, Christian: *Analyse des Zeitverhaltens von Steuerungssystemen mit zyklischer Kommunikation*. In: Kommunikation in der Automation (KommA), S. 7, 2012.

[Horstmann 2005] Horstmann, Marc: *Verflechtung von Test und Entwurf für eine verlässliche Entwicklung eingebetteter Systeme im Automobilbereich*. : Dissertation, Technische Universität Braunschweig, Braunschweig, 2005.

[Hummel 2001] Hummel, Thorsten: *Der Einsatz von hybriden Petri-Netzen für den Entwurf gemischt analog-digitaler eingebetteter Systeme*. In: Dieter Monjau (Hrsg.): Methoden und Beschreibungssprachen zur Modellierung und Verifikation von Schaltungen und Systemen (MBMV), S. 49–58, 2001.

[Hunstock et al. 2000] Hunstock, Ralf; Ruping, Stefan und Rückert, Ulrich.: *A distributed simulator for large networks used in building automation systems*. In: IEEE (Hrsg.): 3rd IEEE International Workshop on Factory Communication Systems. WFCS, S. 203–210, 2000.

[Ichikawa/Hiraishi 1988] Ichikawa, Atsunobu und Hiraishi, Kunihiko: *Analysis and control of discrete event systems represented by petri nets*. In: Varaiya, P.;Kurzhanski, A. B. (Hrsg.): Discrete Event Systems: Models and Applications (Lecture Notes in Control and Information Sciences), S. 115–134. Springer, Berlin/Heidelberg, 1988.

[Iordache/Antsaklis 2006] Iordache, Marian Valentin und Antsaklis, Panos J.: *Supervisory control of concurrent systems. A Petri net structural approach*, Birkhäuser, Boston, 2006.

[Isermann 2005] Isermann, Rolf: *Model-based fault-detection and diagnosis – status and applications*, In: Annual Reviews in Control, 29 (2005) Heft 1, S. 71–85, 2005.

[Jamoussi/Kaminska 1993] Jamoussi, Mohamed und Kaminska, Bozena: *Controllability and observability measures for functional-level testability evaluation*. In: Digest of Papers Eleventh Annual 1993 IEEE VLSI Test Symposium, S. 154–157, 1993.

[Jensen 1992] Jensen, Kurt: *Basic concepts, analysis methods and practical use*. Nr. vol. 1, Springer, Berlin [etc.], 1992.

[Jinrui Nan/Wu Men 2008] Jinrui Nan und Wu Men: *Research on reflection of CAN signal in transmission line*. In: IEEE (Hrsg.): 7th World Congress on Intelligent Control and Automation. WCICA, S. 7707–7710, 2008.

[Jörns 1997] Jörns, Carsten: *Ein integriertes Steuerungsentwurfs- und Verifikationskonzept mit Hilfe interpretierter Petri-Netze* : Dissertation, Universität Kaiserslautern, Kaiserslautern, 1997.

[Kalman 1959] Kalman, Rudolf E..: *On the general theory of control systems,* In: IRE Transactions on Automatic Control, 4 (1959) Heft 3, S. 110, 1959.

[Kastner/Novak 2009] Kastner, Wolfgang und Novak, Thomas: *Functional safety in building automation*. In: Factory Automation (ETFA), S. 1–8, 2009.

[Kiefer 1995] Kiefer, Jörg: *Methodische Partitionierung und Parametrierung von Feldbussen* : Dissertation, Technischen Universität Braunschweig, Braunschweig, 1995.

[Koch et al. 2014] Koch, Stefanie; Bunk, Marcus; Engel, Bernd; Reiss, Dirk; Plesser, Stefan; Fisch, M. Norbert; Diekhake, Patrick und Schnieder, Eckehard: *Das Projekt „Energy Toolkit“ -Dienstleistungen für Kommunen, Netzbetreiber und die Wohnungswirtschaft*. In: VDE (Hrsg.): Smart Cities - Intelligente Lösungen für das Leben in der Zukunft, 2014.

[Kraft 2002] Kraft, Karl Heinz: *Simulation von CAN-Bus-Komponenten und -Übertragungsnetzen*. In: VDE-Fachtagung Analog, 2002.

[Krammer et al. 2010] Krammer, Martin; Clazzer, Federico; Armengaud, Eric; Karner, Michael; Steger, Christian und Weiss, Reinhold: *Exploration of the FlexRay signal integrity using a combined prototyping and simulation approach*. In: IEEE (Hrsg.): 13th Symposium on Design and Diagnostics of Electronic Circuits and Systems. DDECS, S. 111–116, 2010.

[Krause 2012] Krause, Jan: *Testfallgenerierung aus modellbasierten Testfallgenerierung aus modellbasierten Systemspezifikationen auf der Basis von Petrinetzentfaltungen* : Dissertation, Otto-von-Guericke-Universität Magdeburg, Magdeburg, 2012.

[Kupke 2007] Kupke, Torben: *Funkbasierte energieautarke Kommunikation für Eisenbahngüterzüge* : Dissertation, Technische Universität Braunschweig, Braunschweig, 2007.

[Kurschl 2000] Kurschl, Werner: *Monitoring von verteilten Systemen* : Dissertation, Johannes Kepler Universität, Linz, 2000.

[Lauber/Göhner 1999] Lauber, Rudolf und Göhner, Peter: *Automatisierungssysteme und -strukturen, Computer- und Bussysteme für die Anlagen- und Produktautomatisierung, Echtzeitprogrammierung und Echtzeitbetriebssysteme, Zuverlässigkeits- und Sicherheitstechnik*. Nr. / Rudolf Lauber; Peter Göhner, völlig neubearb. Aufl., Springer, Berlin [u.a.], 1999.

[Lefebvre 2014] Lefebvre, Dimitri: *On-Line Fault Diagnosis With Partially Observed Petri Nets*, In: IEEE Transactions on Automatic Control, 59 (2014) Heft 7, S. 1919–1924, 2014.

[Litz 1995] Litz, Lothar: *Entwurf industrieller Prozeßsteuerungen auf der Basis geeigneter Petri-Netz-Interpretationen*. In: Schnieder, E. (Hrsg.): Entwurf komplexer Automatisierungssysteme. EKA, S. 417–429, 1995.

[Litz/Frey 1999] Litz, Lothar und Frey, Georg: *Methoden und Werkzeuge zum industriellen Steuerungsentwurf - Historie, Stand, Ausblick*, In: at - Automatisierungstechnik, 47 (1999) Heft 4, 1999.

[Liu et al. 2013] Liu, Jieyu; Schnieder, Eckehard und Miao, Zuoyu: *Message Collision and CSMA/CA Modeling and Simulation by Means of Petri Net*. In: IEEE (Hrsg.): International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery. CyberC, S. 413–420, 2013.

[Lunze 2012] Lunze, Jan: *Automatisierungstechnik. Methoden für die Überwachung und Steuerung kontinuierlicher und ereignisdiskreter Systeme*. 3., überarb. Auflage. Aufl., Oldenbourg, München, 2012.

[Marsal 2008] Marsal, Gaëlle: *Bewertung des Zeitverhaltens Ethernet-basierter Automatisierungssysteme durch Simulation mit höheren Petrinetzen*, In: at - Automatisierungstechnik, 56 (2008) Heft 4, 2008.

[McCammon 2011a] McCammon, Roy: *SPICE Simulation of Transmission Lines by the Telegrapher's Method. Putting the Telegrapher's Equations into a Usable Sub-Circuit*. <http://i.cmpnet.com/rfdesignline/2010/06/C0580Pt3edited.pdf>, Rev. 2011-11-18, Stand 2015-01-30.

[McCammon 2011b] McCammon, Roy: *SPICE Simulation of Transmission Lines by the Telegrapher's Method. Putting the Telegrapher's Equations into a Circuit*. <http://i.cmpnet.com/rfdesignline/2010/06/C0580Pt1edited.pdf>, Rev. 2011-11-18, Stand 2015-01-30.

[Meda-Campana et al. 2000] Meda-Campana, M. E.; Ramirez-Trevino, A. und Lopez-Mellado, E.: *Dynamical local properties for estimation and control of discrete event systems modeled by interpreted Petri nets*. In: IEEE (Hrsg.): International Conference on Systems, Man, and Cybernetics. SMC, S. 2150–2155, 2000.

[Meyer zu Hörste 2004] Meyer zu Hörste, Michael zu Hörste: *Methodische Analyse und generische Modellierung von Eisenbahnleit- und Sicherungssystemen* : Dissertation, Technische Universität Braunschweig, Braunschweig, 2004.

[Muller/Valle 2010] Muller, Candice und Valle, Maurizio: *System verification of flexray communication networks through behavioral simulations*. In: IEEE International Behavioral Modeling and Simulation Conference (BMAS), S. 1–6, 2010.

[Neumann et al. 2011] Neumann, Christian; Jacob, Dirk; Burhenne, Sebastian; Florita, Anthony und Burger, Eberhard: *ModBen Modellbasierte Methoden für die Fehlererkennung und Optimierung im Gebäudebetrieb* : Endbericht zum Projekt ModBen, Förderkennzeichen: 0327410A-C, 2011.

[Nguyen et al. 2008] Nguyen, Thang; Haase, Joachim und Pelz, Georg: *Sensitivity analysis of passive CAN bus components to investigate signal integrity of CAN network physical layer*. In: IEEE (Hrsg.): International Behavioral Modeling. BMAS, S. 55–60, 2008.

[Ober 1999] Ober, Bernhard: *Modellgestützte Synthese ereignisdiskreter Steuerungen* : Dissertation, Technische Universität Braunschweig, Braunschweig, 1999.

[Oberquelle 1981] Oberquelle, Horst;, Communication by graphic net representations, Bericht Nr.75 des Fachbereichs Informatik, Universität Hamburg, 1981.

[Page/Liebert 1991] Page, Bernd und Liebert, Hansjörg: *Diskrete Simulation. Eine Einführung mit Modula-2*, Springer, Berlin [u.a.], 1991.

[Plesser et al. 2012] Plesser, Stefan; Claas, Pinkernell und Rehbein, Christoph: *Aktive Funktionsbeschreibungen. Software zeigt Betriebsdefizite und definiert Qualitätsstandards*. In: Bauverlag (Hrsg.): TAB Technik am Bau, S. 46–49, 2012.

[Pohlmann 2014] Pohlmann, Norbert: *IT-Sicherheit in intelligenten Gebäuden*, In: Loytec Express (2014), S. 12–15, 2014.

[Polke 1994] Polke, Martin: *Prozessleittechnik*, 2., überarb. Aufl., Oldenbourg Wissenschaftsverlag, 1994.

[Prodanov et al. 2009] Prodanov, William; Valle, Maurizio und Buzas, Roman: *A Controller Area Network Bus Transceiver Behavioral Model for Network Design and Simulation*, In: IEEE Transactions on Industrial Electronics, 56 (2009) Heft 9, S. 3762–3771, 2009.

[Quiroga et al. 2014] Quiroga, Lisandro; Becker, Uwe und Schnieder, Eckehard: *Das Petrinetz Modellierungs- und -analysetool Pi-Tool*. In: De Gruyter (Hrsg.): at-Automatisierungstechnik, S. 436–445, 2014.

[Ramadge/Wonham 1989] Ramadge, Peter und Wonham, Murray: *The control of discrete event systems*. In: Proc. IEEE Vol. 77, 1989.

[Ramirez-Trevino et al. 2003] Ramirez-Trevino, A.; Rivera-Rangel, I. und Lopez-Mellado, E.: *Observability of discrete event systems modeled by interpreted petri nets*, In: IEEE Transactions on Robotics and Automation, 19 (2003) Heft 4, S. 557–565, 2003.

[Reisig 1982] Reisig, Wolfgang: *Petrinetze. Eine Einführung*. Springer, Berlin, 1982.

[Reisig 2013] Reisig, Wolfgang: *Understanding Petri Nets. Modeling Techniques, Analysis Methods, Case Studies*. Springer, Berlin, 1982.

[Rivera-Rangel et al. 2005] Rivera-Rangel, I.; Ramirez-Trevino, A.; Aguirre-Salas, Luis und Ruiz-Leon, J.: Geometrical characterisation of observability in interpreted Petri nets, In: Kybernetika (2005) Heft 41, S. 553–574, 2005.

[Ropohl 2012] Ropohl, Günter: *Allgemeine Systemtheorie- Einführung in transdisziplinäres Denken*. edition sigma, Berlin, 2012.

[Runde et al. 2011] Runde, Stefan; Fay, Alexander; Schmitz, Stefan und Epple, Ulrich: *Wissensbasierte Systeme im Engineering der Automatisierungstechnik*, In: at - Automatisierungstechnik, 59 (2011) Heft 1, 2011.

[Runde 2011] Runde, Stefan: *Wissensbasiertes Engineering in der Automatisierungstechnik am Beispiel der Gebäudeautomatisierung* : Dissertation, Helmut Schmidt Universität, Universität der Bundeswehr, Hamburg, 2011.

[Ruzicka 2003] Ruzicka, Richard: *Testable design verification using Petri nets*. In: Proceedings. Euromicro Symposium on Digital System Design, S. 304–311, 2003.

[Saito 2002] Saito, Seiichi: *A novel analysis method of bus signal transmission and a proposal for high-speed low-power bus circuit.* In: IEEE International Symposium 26-29 May, S. I-89, 2002.

[Saki et al. 2011] Saki, Mehdi; Fereidunian, Alireza; Lesani, Hamid und Kolarijani, Mohamad Amin Sharifi: *Distribution automation monitoring using Petri nets*. In: 2nd International Conference on Control, Instrumentation, and Automation (ICCIA), S. 56–61, 2011.

[Sallans et al. 2006] Sallans, Brian; Bruckner, Dietmar und Russ, Gerhard: *Statistical Detection of Alarm Conditions in Building Automation Systems*. In: 2006 IEEE International Conference, S. 257–262, 2006.

[Schäuffele/Zurawka 2013] Schäuffele, Jörg und Zurawka, Thomas: *Automotive Software Engineering. Grundlagen, Prozesse, Methoden und Werkzeuge effizient einsetzen*. 5., überarb. Aufl., Springer Vieweg, Wiesbaden, 2013.

[Schneider 1997] Schneider, Hans-Jochen: *Lexikon Informatik und Datenverarbeitung*. 4., aktualisierte und erw. Aufl., R. Oldenbourg, München, 1997.

[Schneeweiss 1999] Schneeweiss, Winfrid G.: *Petri Nets for Reliability Modeling*, LiLoLe, Hagen, 1999.

[Schnieder 1993] Schnieder, Eckehard: *Prozessinformatik*. 2., erw. Aufl., Vieweg, Braunschweig, Wiesbaden, 1993.

[Schnieder 1999] Schnieder, Eckehard: *Methoden der Automatisierung. Beschreibungsmittel, Modellkonzepte und Werkzeuge für Automatisierungssysteme*, Vieweg, Braunschweig, Wiesbaden, 1999.

[Schnieder 2007] Schnieder, Eckehard: *Verkehrsleittechnik. Automatisierung des Straßen- und Schienenverkehrs*, Springer, Berlin [u.a.], 2007.

[Schnieder 2010] Schnieder, Lars: *Formalisierte Terminologien technischer Systeme und ihrer Zuverlässigkeit* : Dissertation, Technische Universität Braunschweig, Braunschweig, 2010.

[Schnieder/Chouikha 1998] Schnieder, Eckehard und Chouikha, Mourad: *Modellbasierter Steuerungsentwurf mit Petrinetzen*. In: VDI Bericht 1397 (Hrsg.): Meß- und Automatisierungstechnik. VDI/VDE-GMA, S. 319–326, 1998.

[Schnieder/Kraft 1980] Schnieder, Eckehard und Kraft, Karl Heinz: *Berechnung der Reaktionszeiten in Steuerungssystemen mit verteilten Elementen*. In: Siemens Forschungs- und Entwicklungsberichte. 9(6), S. 325–329, 1980.

[Schnieder/Schnieder 2012] Schnieder, Eckehard und Schnieder, Lars: *Axiomatik der Begriffe für die Automatisierungstechnik*. In: Deutscher Industrieverlag (Hrsg.): atp-Automatisierungstechnische Praxis, S. 62–73, 2012.

[Schnieder/Schnieder 2010] Schnieder, Eckehard und Schnieder, Lars: *Terminologische Präzisierung des Systembegriffs*. In: Deutscher Industrieverlag (Hrsg.): atp-Automatisierungstechnische Praxis (52), S. 46–59, 2010.

[Schrom 2003] Schrom, Harald: *Realisierung eines optimierten Feldbussystems und Modellierung mit Petrinetzen* : Dissertation, Technische Universität Braunschweig, Braunschweig, 2003.

[Schrom 2007] Schrom, Harald: *Optimiertes Feldbussystem. Entwurf und Realisierung eines integralen Low-Power*. 1. Aufl. Aufl., VDM Verlag Dr. Müller, Saarbrücken, 2007.

[Schrom et al. 2012] Schrom, Harald; Michaels, Tobias; Klein, Ralf; Becker, Uwe; Wegele, Stefan und Quiroga, Lisandro: *EkReit. Embedded Systems mit kleinen Recheneinheiten und zuverlässigem Zeitverhalten* : Endbericht zum Projekt EkReit, Förderkennzeichen: 01IS09039A-C, 2012.

[Singh et al. 2012] Singh, Lalit Kumar; Vinod, Gopika und Tripathi, A. K.: *Modeling and Prediction of Performability of Safety Critical Computer Based Systems Using Petri Nets*. In: IEEE (Hrsg.): International Symposium on Software Reliability Engineering Workshops. ISSREW, S. 85–94, 2012.

[Starke 1990] Starke, Peter H.: *Analyse von Petri-Netz-Modellen*. Springer, Berlin 1990.

[Stein et al. 2010] Stein, Christian; Schnieder, Lars und Schielke, Arno: *Effektives Terminologiemanagement mit dem Werkzeug IGLOS*. In: Entwurf komplexer Automatisierungssysteme (EKA), 2010.

[Störtkuhl 2014] Störtkuhl, Thomas: *Security analysis for electric signalling systems*. In: Formal Methods for Automation and Safety in Railway and Automotive Systems (FORMS), S. 177–183, 2014.

[Tan et al. 1990] Tan, X.; Jue, S. C.; Lo, E. und Hardy, R.H.S.: *Transient performance of a CSMA system under temporary overload conditions*. In: 15th Conference on Local Computer Networks, S. 93–101, 1990.

[Tanenbaum/van Steen 2008] Tanenbaum, Andrew S. und van Steen, Maarten: *Verteilte Systeme. Prinzipien und Paradigmen*. 2., aktualisierte Aufl., Pearson Studium, München [u.a.], 2008.

[Tari/Bukhres 2001] Tari, Zahir und Bukhres, Omran: *Fundamentals of distributed object systems. The CORBA perspective*, Wiley, New York, 2001.

[Tianjing Jiang et al. 2000] Tianjing Jiang; Klenke, Robert H.; Aylor, James H. und Gang Han: *System level testability analysis using Petri nets*. In: International High-Level Design Validation and Test Workshop, S. 112–117, 2000.

[Wang et al. 2012] Wang, Ruomin; Denoulet, Julien; Feruglio, Sylvain; Vallette, Farouk und Garda, Patrick: *High level modeling of signal integrity in field bus communication with SystemC-AMS*. In: 19th IEEE International Conference on Electronics, Circuits and Systems - (ICECS), S. 889–892, 2012.

[Werthschulte 2003] Werthschulte, Kay: Integration von heterogenen Bussystemen in die Heimautomatisierung unter Verwendung von Middleware : Dissertation, Technischen Universität München, München, 2003.

[Yang Yang/Tak-Shing Peter Yum 2003] Yang Yang und Tak-Shing Peter Yum: *Delay distributions of slotted ALOHA and CSMA*, In: IEEE Transactions on Communications, 51 (2003) Heft 11, S. 1846–1857, 2003.

[Zdenek/Jiri 2013] Zdenek, Kubík und Jiri, Skala: *Simulation of CAN Bus Physical Layer Using SPICE*. In: Pinker, J. (Hrsg.): 2013 International Conference on Applied Electronics. Pilsen, 10-12 September 2013, S. 1–4. University of West Bohemia, Pilsen, 2013.

[Zhu 2014] Zhu, Guowen: *Hierarchische Überwachung eines verteilten Automatisierungssystems mit Petrinetzen* : Studienarbeit, Technische Universität Braunschweig, Braunschweig, 2014.

[Zimmermann 1980] Zimmermann, Hubert: *OSI Reference Model-The ISO Model of Architecture for Open Systems Interconnection*, In: IEEE Transactions on Communications, 28 (1980) Heft 4, S. 425–432, 1980.

[Zimmermann/Schmidgall 2006] Zimmermann, Werner und Schmidgall, Ralf: Bussysteme in der Fahrzeugtechnik. Protokolle und Standards. 1. Aufl., Vieweg, Wiesbaden, 2006.

Abbildungsverzeichnis

Abbildung 1.1: Ausschnitt aus der manuell erstellten Systemarchitektur eines durch SmallCAN gesteuerten Büroraumes 7

Abbildung 1.2: Zustandsgraph (oben) und Systemverhalten (unten) der Funktion zur Belegungsauswertung.....10

Abbildung 1.3: Vorherrschende Analysetätigkeiten im Lebenszyklus eines Automatisierungssystems.....17

Abbildung 2.1: Darstellung des Lebenszyklus eines Automatisierungssystems und exemplarisch skizzierte Ansätze zur Vereinheitlichung der Analysetätigkeiten.....25

Abbildung 2.2: Konzept zur vereinheitlichten Analyse eines Automatisierungssystems.....26

Abbildung 2.3: Netzmodelle für die Analyse am Beispiel der Funktion einer Raumbelegungsauswertung.....30

Abbildung 3.1: Allgemeiner Systembegriff und seine Eigenschaften [Schnieder 2010] 37

Abbildung 3.2: Konkretisierung vom Systembegriff und exemplarische hierarchische Beschreibungsstruktur am Beispiel der Struktur eines technischen Systems und des Transportprozesses.....37

Abbildung 3.3: Darstellung der dekomponierten Subsysteme eines verteilten Automatisierungssystems.....40

Abbildung 3.4: Ausschnitt aus dem Hierarchieschema zur Beschreibung des funktionalen Anwendungssystems eines verteilten Automatisierungssystems und der Raumautomation als dessen Konkretisierung.....41

Abbildung 3.5: Abstrahierter Zustandsgraph der implementierten Raumautomation (in Anlehnung an die Richtlinie [VDI 3814:2009]) 42

Abbildung 3.6: Ausschnitt aus dem Hierarchieschema zur Beschreibung des Bitübertragungssystems am Beispiel von SmallCAN.....46

Abbildung 3.7: Ausschnitt aus dem Hierarchieschema zur Beschreibung des Datenübertragungssystems am Beispiel von SmallCAN 48

Abbildung 3.8: Ausschnitt aus dem Hierarchieschema zur Beschreibung von SmallCAN-Ressourcen zur Realisierung einer Außenluftregelung.....50

Abbildung 3.9: Ausschnitt aus dem Hierarchieschema zur Beschreibung einer Ressource am Beispiel eines einfachen Telefonkabels als Busmedium 51

Abbildung 3.10: Ausschnitt aus dem Hierarchieschema zur Beschreibung von Anforderungen an die betrachteten Systeme.....53

Abbildung 3.11: Ausschnitt aus einem ganzheitlichen Systemmodell mit Beschreibung eines funktional betrachteten Anwendungssystems (rot) und einer SmallCAN-Raumautomation als dessen Instanz.....56

Abbildung 3.12: Ausschnitt aus dem ganzheitlichen Systemmodell zur Beschreibung des Datenübertragungssystems als Dekomposition vom System SmallCAN.....57

Abbildung 4.1: Ganzheitliches Systemmodell, angepasst an die SmallCAN-spezifischen Problemstellungen zur elektromagnetischen Verträglichkeit und zur Signalqualität, mit räumlich technologisch betrachteten Bitübertragungssystem (braun) als Dekomposition von SmallCAN 63

Abbildung 4.2: Deskriptives Modell eines minimalen Aufbaus und beschreibende Netzliste in SPICE.....64

Abbildung 4.3: Deskriptives Modell eines maximalen Aufbaus (oben) und beschreibende Netzliste in SPICE (unten) 66

Abbildung 4.4: Vereinfachte Betrachtung des Bitübertragungssystems mittels Reihenschwingkreis 67

Abbildung 4.5: Signalverhalten nach dem Schalten des Reihenschwingkreises für verschiedene Parameterwerte.....68

Abbildung 4.6: Beschreibung der Leitung über periodisch angeordnete Vierpole diskreter Bauelemente 69

Abbildung 4.7: Beschreibung der Leitung über homogen verlaufende Bauteilkomponenten.....69

Abbildung 4.8: Deskriptives Modell eines zu validierenden Aufbaus mit 5 Transceivern und einer Gesamtleitungslänge von 900 m.....71

Abbildung 4.9: Fallende Flanken (links) und steigende Flanken (rechts) im zeitlichen Signalverlauf, gemessen (oben) und simuliert (unten) bei aktiven Transceiver TR_3.....72

Abbildung 4.10: Simulationsergebnisse für eine minimale (oben) und eine maximale (unten) Topologie bei ungesteuertem Schalten des Ausgangstransistors ..75

Abbildung 4.11: Simulationsergebnisse für eine minimale (oben) und eine maximale (unten) Topologie bei gesteuertem Schalten des Ausgangstransistors 76

Abbildung 4.12: Fallende Flanke im zeitlichen Signalverlauf simuliert nah an der Versorgungseinheit und am sendenden Buskoppler (oben) sowie am Ende des Netzwerkes (unten) für unterschiedlich ausgedehnte Bustopologien ..77

Abbildung 4.13: Rückschwingverhalten bei unterschiedlich ausgeprägten Stichleitungslängen, simuliert in der Nähe des sendenden Buskopplers ...78

Abbildung 4.14: Einfluss von CAN-Drosseln als Filtermaßnahme auf die Störaussendung, gemessen in einer TEM-Zelle.....80

Abbildung 4.15: Einfluss verschiedener Schirmmaßnahmen auf die Störaussendung, gemessen in einer TEM-Zelle 81

Abbildung 4.16: Störaussendungsmessung in Modenverwirbelungskammer für verschiedene Ansteuerungsarten des Ausgangstransistors 82

Abbildung 4.17: Normative Messung zur Störaussendung in Schirmkabine für verschiedene Ansteuerungsarten des Ausgangstransistors 82

Abbildung 4.18: Frequenzgang-Darstellung simulierter Signalverläufe für die verschiedenen Ansteuerungen des Ausgangstransistors 83

Abbildung 4.19: Ergebnisrückführung in das Systemmodell nach der Signalqualitätsanalyse 85

Abbildung 4.20: Ergebnisrückführung in das Systemmodell nach den Analysen zur elektromagnetischen Verträglichkeit.....85

Abbildung 5.1: Ganzheitliches Systemmodell angepasst an die Problemstellung des Buslastverhaltens in einem verteilten Automatisierungssystem mit funktional betrachteten Kommunikationssystem (blau) und funktional betrachteten Anwendungssystem (rot) als Dekompositionen eines verteilten Automatisierungssystems.....88

Abbildung 5.2: Abbildung des funktional betrachteten Kommunikationssystems (blau), gekoppelt an das Ressourcenmodell eines Bussystems bestehend aus fünf Gerätekomponenten am Beispiel von SmallCAN.....92

Abbildung 5.3: Verzögerungszeit eines niederprioren Telegrammes bei steigender Busauslastung von $\underline{\underline{A}}_{K,0\%}$ bis $\underline{\underline{A}}_{K,100\%}$ (links) und Verteilung der Verzögerungszeit bei maximaler Busauslastung $\underline{\underline{A}}_{K,100\%}$ (rechts).....94

Abbildung 5.4: Simulierte Kommunikationsverzögerungen eines niederprioren Telegrammes bei unterschiedlichen Busauslastung.....95

Abbildung 5.5: Einfluss der Versandart auf die Verzögerungszeiten von Telegrammen niedriger und mittlerer Priorität bei maximaler Busauslastung.....96

Abbildung 5.6: Einfluss der Teilnehmeranzahl auf die Verzögerungszeiten bei maximaler Busauslastung bei deterministischer und stochastischer Versandart 97

Abbildung 5.7: Ganzheitliches Systemmodell mit funktional und technologisch zu betrachteten Anwendungssystem (violett) angepasst an die Problemstellung des Zyklusübergangs- und Synchronisationsverhalten in einem verteilten Automatisierungssystem 99

Abbildung 5.8: Petrinetzmodell eines linear verknüpften Verbundes mit sieben Programmkomponenten 102

Abbildung 5.9: Verzögerungsverhalten an den Programmübergängen eines linearen Verbundes mit sieben Programmkomponenten verschiedener Zykluszeiten 105

Abbildung 5.10: Simulierter zeitliche Verlauf der Verzögerungszeiten an den Programmübergängen eines linearen Verbundes mit sieben Programmkomponenten verschiedener Zykluszeiten 106

Abbildung 5.11: Petrinetzmodell eines parallelen Verbundes mit sechs Programmkomponenten und deren Synchronisation an der siebten Programmkomponente 108

Abbildung 5.12: Verzögerungsverhalten durch die Synchronisation eines parallelen Verbundes bei deterministischen Funktionsaufrufen 109

Abbildung 5.13: Verzögerungsverhalten durch die Synchronisation eines parallelen Verbundes bei stochastischen Funktionsaufrufen 110

Abbildung 5.14: Systemmodell angepasst an die Problemstellung des Laufzeitverhaltens in einem verteilten Automatisierungssystem mit Anwendungssystem (cyan) und abstrahierten kausalen Ablauf (gelb) 113

Abbildung 5.15: Funktionverknüpfungen der Raumautomation und exemplarisch hervorgehobene Funktionskette (rot dargestellt).....117

Abbildung 5.16: Petrinetzmodell des Gesamtprozesses (cyan) und eines ausgewählten zu analysierenden Teilprozesses (gelb) aus einer Schaltfolge (rot) unter Berücksichtigung von Zyklusübergängen und Kommunikationsprozessen 118

Abbildung 5.17: Exemplarischer kausaler Prozessablauf unter Berücksichtigung von Synchronisations- und Kommunikationsprozessen 120

Abbildung 5.18: Simuliertes Laufzeitverhalten (links) einer Schaltfolge von Prozesschritten und vereinfachte Darstellung als homogener Poissonprozess (rechts) bei minimaler (oben) und maximaler (unten) Busauslastung.....122

Abbildung 5.19: Ergebnisrückführung in das Systemmodell nach den Analysen zur Zeitbewertung.....123

Abbildung 6.1: Systemmodell zur Darstellung der in diesem Kapitel behandelten Beobachtbarkeits-und Steuerbarkeitsanalysen an den Beispielen der funktional betrachteten Funktion zur Belegungsauswertung (grün) und eines funktional räumlich betrachteten Teilprozesses (orange) bzw. eines abstrahierten kausalen Prozesses (dunkelrot).....127

Abbildung 6.2: Interpretiertes Petrinetz eines exemplarischen Steuerungsprozesses.131

Abbildung 6.3: Exemplarischer Steuerungsprozess mit Kontrollstruktur.....132

Abbildung 6.4: Petrinetz zur Analyse der Sequenzdetektierbarkeit eines Prozessablaufes mit nicht detektierbaren Zuständen 137

Abbildung 7.1: Einlesen und binäres Auswerten eines Telegrammes zur Schaltbeeinflussung von Transitionen eines Petrinetzes mit externen Stellen.....142

Abbildung 7.2: Externe Stelle mit zugewiesenem Funktionsaufruf zur Telegrammaussendung.....143

Abbildung 7.3: Systemmodell zur Darstellung der in diesem Abschnitt herangezogenen Analysemodelle für die Online-Analyse am Beispiel einer Testausführung (lila) und einer Laufzeitprüfung (hellgrün) 144

Abbildung 7.4: Petrinetzmodell zur Ausführung eines Testfalles auf Basis des Modellkonstruktes eines Testablaufes (lila) 145

Abbildung 7.5: Prozess für die Durchführung eines exemplarischen Testfalls für die Funktion der Belegungsauswertung 149

Abbildung 7.6: Petrinetzmodell zur Laufzeitprüfung eines Steuerungsprozesses auf Basis des Modellkonstruktes (hellgrün) zur Erfassung des Realverhaltens des Steuerungssystems 151

Abbildung 7.7: Soll-Modell zur Laufzeitüberwachung eines exemplarischen Prozessablaufes 158

Abbildung 7.8: Anwendbares Modell zur Laufzeitüberwachung eines exemplarischen Prozessablaufes 159

Abbildung A.1: Zustandsgraph des Außenluftreglers 189

Abbildung A.2: Geometrische Anordnung der Bauteile des Anwendungsmoduls 200

Tabellenverzeichnis

Tabelle 1: Zuordnung von SmallCAN-Ressourcen zu Funktionen aus [VDI 3813:2011] 5

Tabelle 2: Zuordnungstabelle für die Funktion zur Belegungsauswertung 9

Tabelle 3: Darstellung von Problemen, Anforderungen und Lösungsansätzen verteilter (Gebäude-)Automatisierungssysteme 16

Tabelle 4: Gliederung und Fokus dieser Arbeit 23

Tabelle 5: Modellkonzepte verschiedener Beschreibungsmittel 32

Tabelle 6: Geeignete Werkzeuge für die Modellierung und Simulation von Petrinetz-Modellen [Quiroga et al. 2014] 33

Tabelle 7: Geeignete Werkzeuge für die Modellierung und Simulation elektrischer Schaltungen 34

Tabelle 8: Kenngrößen von physikalischen Netztopologien und deren Beeinflussung auf ausgewählte Qualitätsmerkmale 45

Tabelle 9: Merkmale verschiedener Bitübertragungssysteme 59

Tabelle 10: Maßnahmen zur Erhöhung der Störfestigkeit des SmallCAN-Buskopplers 79

Tabelle 11: Bewertete Maßnahmen zur Reduzierung der Störaussendung für SmallCAN 84

Tabelle 12: Minimale und maximale ermittelte Laufzeiten des kausalen Prozessablaufes unter Berücksichtigung sämtlicher Verzögerungszeiten bei minimaler und maximaler Busauslastung 121

Anhang A

Realisierung einer Außenluftregelung

A.1 Beschreibung der Systemfunktion

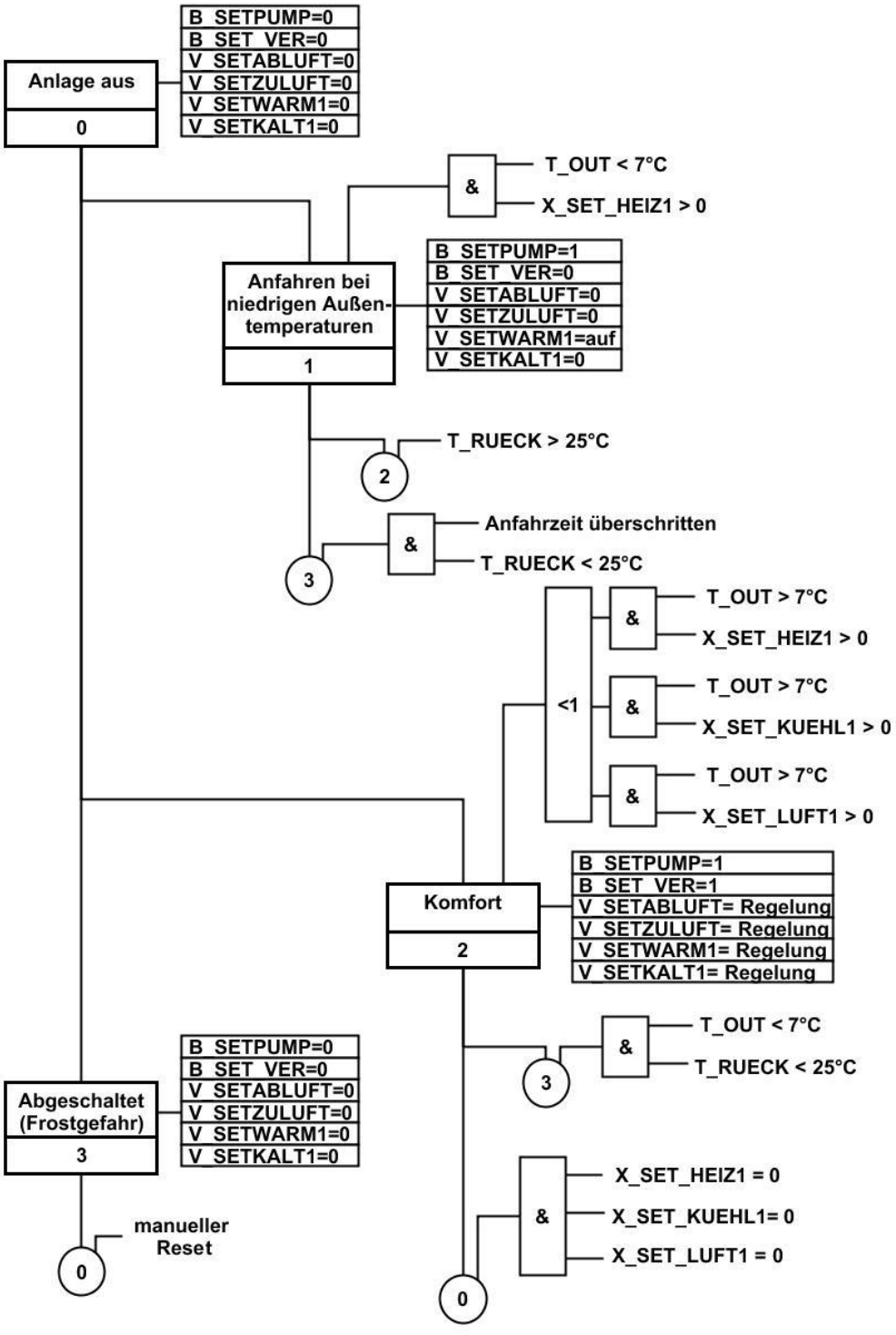


Abbildung A.1: Zustandsgraph des Außenluftreglers

A.2 Programmcode

```
// ********************************************************************
// *** freie Sonderfunktion
// *** AUSLG_FSF.c
// *** Außenluftgeraet
// *** Diese Funktion liest zahlreiche Eingangswerte und steuert
// *** auf dieser Basis ein Außenluftgerät, das aus mehreren
// *** Buskopplern besteht. (Future Workspace, Trox-Einbaugeraet)
// ********************************************************************
//
#define     C_BEZ1                      'A'             // [A..Z, 0..4, SPCAE] Bezeichnung der Sonderfunktion
#define     C_BEZ2                      'U'             // [A..Z, 0..4, SPCAE]
#define     C_BEZ3                      'S'             // [A..Z, 0..4, SPACE]
#define     C_BEZ4                      'L'             // [A..Z, 0..4, SPACE]
#define     C_BEZ5                      'G'             // [F..T,       SPACE]

#define     C_VERS                      1               // Versionsnummer der Sonderfunktion
//*********************************************************************
//*** INCLUDES
//*********************************************************************
                    // 16F88.h wird durch C-Compiler included
                    #include    "FSFc.h"
//*********************************************************************
//*** Bibliotheken
//*********************************************************************
extern page1 uns8 L_LIB_READEE (uns8);
//*********************************************************************
//**** Genutzte Stacktiefe                          ****
//*********************************************************************
//
            //L_INIT_FREI_0             ?
            //L_START_FREI_0            ?
            //L_ZYKL_LANGSAM_FR_0       ?
            //L_T13_FREI_0              ?
            //L_T20_FREI_0              ?
            //L_T01_FREI_0              ?
            //L_SEND_FREI1_0            ?
            //L_SEND_FREI1_ENDE_0       ?
//*********************************************************************
//*** Prototypen
//*********************************************************************
//
//*********************************************************************
//**** Variable/Konstante/Adressen der Sonderfunktion ***************
//*********************************************************************
//**** Adressen ***********************************************************
#define     EE_HeizVentilModus                  255U
#define     EE_KuehlVentilModus                 254U
#define      EE_UntererSkalierungsWert          253U
#define     EE_Skalierungsfaktor                251U
#define     EE_Anfahrzeit                       250U
#define     EE_SkalierungsfaktorAbluft          249U

//**** Konstante **********************************************************
// Minimale Ruecklauftemperatur um bei Frostschutz ein korrektes Anfahren zu erkennen (in 0.1 °C)
#define Min_RL_Temp_Anfahren            100U


//**** Variable ***********************************************************
                                                    // 0x190, 96 Byte in Bank 3
uns8    SEND_FLAGS;                                 // ID-Blockversand Anforderung
uns8    SEND_TXT;                                   // ID-Blockversand im Versand
uns8    Kuehlanforderung;
uns8    Heizanforderung;
uns8    Luftwechselrate_Anforderung;
uns8    Luefter_Abluft;
uns8    Luefter_Zuluft;
uns8    HeizVentilStellung;
uns8    KuehlVentilStellung;
uns8    HeizVentilModus;
uns8    KuehlVentilModus;
uns8    Skalierungsfaktor;
```

```
uns8   SkalierungsfaktorAbluft;
uns8   UntererSkalierungsWert;
uns16  SkalierterAusgang;
int16  WW_Ruecklauftemperatur;
int16  Aussen_Temperatur;
uns8   SekundenCounter;
uns8   Anfahrzeit;
uns8   Versuche_Anfahrschaltung;
uns8   Flags;
bit    NewValue@Flags.0;
bit    Heizen@Flags.1;
bit    Kuehlen@Flags.2;
bit    Anforderung_Luft_An@Flags.3;
bit    Pumpe_An@Flags.4;
bit    Luefter_An@Flags.5;
bit    ByPass_Aktiv@Flags.6;
bit    Zustand_Pumpe@Flags.7;
uns8   FrostschutzFlags;
bit    Frostschutz_Fehler@FrostschutzFlags.0;
bit    Frostschutz@FrostschutzFlags.1;
bit    Anfahrschaltung_Start@FrostschutzFlags.2;
bit    Anfahrschaltung_Aktiv@FrostschutzFlags.3;
bit    Starte_Anfahrschaltung@FrostschutzFlags.4;
bit    Anfahrschaltung_Erfolgreich@FrostschutzFlags.5;
uns8    temp;                                          // Zur Empfangsdatenunterscheidung und für andere temporäre
                                                       Daten
//
//**** LIB-Variable ************************************************
//
//**** Flags *******************************************************
#define FL_SEND_PUMPE              SEND_FLAGS.0     // Sub-ID 0
#define FL_SEND_POWER              SEND_FLAGS.1     // Sub-ID 1
#define FL_SEND_ABLUFT             SEND_FLAGS.2     // Sub-ID 2
#define FL_SEND_ZULUFT             SEND_FLAGS.3     // Sub-ID 3
#define FL_SEND_MODHEIZ            SEND_FLAGS.4     // Sub-ID 4
#define FL_SEND_MODKUHL            SEND_FLAGS.5     // Sub-ID 5
#define FL_SEND_VENHEIZ            SEND_FLAGS.6     // Sub-ID 5
#define FL_SEND_VENKUHL            SEND_FLAGS.7     // Sub-ID 5
#define MASK_SEND_FSFALL 0b11111111                  // Maske fuer alle Sub-IDs

#define FL_SENDTXT_PUMPE           SEND_TXT.0       // Sub-ID 0
#define FL_SENDTXT_POWER           SEND_TXT.1       // Sub-ID 1
#define FL_SENDTXT_ABLUFT          SEND_TXT.2       // Sub-ID 2
#define FL_SENDTXT_ZULUFT          SEND_TXT.3       // Sub-ID 3
#define FL_SENDTXT_MODHEIZ         SEND_TXT.4       // Sub-ID 4
#define FL_SENDTXT_MODKUHL         SEND_TXT.5       // Sub-ID 5
#define FL_SENDTXT_VENHEIZ         SEND_TXT.6       // Sub-ID 6
#define FL_SENDTXT_VENKUHL         SEND_TXT.7       // Sub-ID 7


//*****************************************************************
//**** Subroutine
//*****************************************************************
page1 void setVentilatoren (uns8 temp)
{
   SkalierterAusgang = (uns16) Skalierungsfaktor * temp;
   SkalierterAusgang = SkalierterAusgang>>8;                  // Skalierungsfaktor ist als < 1 aufzufassen
   SkalierterAusgang = SkalierterAusgang + UntererSkalierungsWert;
   if (SkalierterAusgang>255)
     SkalierterAusgang=255;
   Luefter_Zuluft = SkalierterAusgang;
   SkalierterAusgang = (uns16) SkalierungsfaktorAbluft * temp;
   SkalierterAusgang = SkalierterAusgang>>8;                  // Skalierungsfaktor ist als < 1 aufzufassen
   SkalierterAusgang = SkalierterAusgang + UntererSkalierungsWert;
   if (SkalierterAusgang>255)
     SkalierterAusgang=255;
   Luefter_Abluft = SkalierterAusgang;

   FL_SEND_ABLUFT = 1;
   FL_SEND_ZULUFT = 1;
   FLS_FREI1 = 1;
}

#pragma stackLevels 4
extern page1 void L_ZYKL_LANGSAM_FR_0 (void)
#define     ZYKL_LANGSAM_FR            1
```

```
{
 if (!(NewValue))                                    //  Nur rechnen, wenn neue Daten eingegangen sind
   return;
 NewValue=0;

//*****************************************************************************************************************
*************************************************************************

 if ( Frostschutz_Fehler==1)                          // Wenn 3x Anfahrschaltung fehlgeschlagen
                                                      -> Lüfter ausschalten  -> Klappe schließt
 {
   if (Pumpe_An)                                      // Wenn die Pumpe noch an ist, dann abschalten
    {
     Pumpe_An=0;
     FL_SEND_PUMPE = 1;
    }
    if (Luefter_An)                                   // Wenn die Lüfter noch an sind, dann abschalten
    {
     Luefter_An=0;
     FL_SEND_POWER = 1;
    }
    if ((Luefter_Abluft>0) || (Luefter_Zuluft>0))     // Wenn die Lüfterdrehzahl noch auf >0 eingestellt ist,
                                                      dann auf 0 stellen
    {
     Luefter_Abluft = 0;
     Luefter_Zuluft = 0;
     FL_SEND_ABLUFT = 1;
     FL_SEND_ZULUFT = 1;
    }
    if ((HeizVentilStellung>0) || (HeizVentilModus!=0))  // Wenn das Heizventil noch offen ist, oder der
                                                      Heizventilmodus nicht auf Durchgriff steht, dann
                                                      Modus auf Durchgriff und Heizventil schließen
    {
     HeizVentilModus=0;
     HeizVentilStellung = 0;
     FL_SEND_MODHEIZ = 1;
     FL_SEND_VENHEIZ = 1;
    }
    if ((KuehlVentilStellung>0) || (KuehlVentilModus!=0))  // Wenn das Kuehlventil noch offen ist, oder der
                                                      Kuehlventilmodus nicht auf Durchgriff steht, dann
                                                      Modus auf Durchgriff und Kuehlventil schließen
    {
     KuehlVentilModus=0;
     KuehlVentilStellung = 0;
     FL_SEND_MODKUHL = 1;
     FL_SEND_VENKUHL = 1;
    }
   FLS_FREI1=1;
   return;
 }

//*****************************************************************************************************************
 if (Aussen_Temperatur<70)                            //  Außentemperatur kleiner 7°C -> Frostschutz aktiv?
    Frostschutz=1;
 else
    Frostschutz=0;

//*****************************************************************************************************************
                                                      //  Falls Bedingungen nicht mehr eingehalten werden können
 if ((WW_Ruecklauftemperatur<=Min_RL_Temp_Anfahren) && (Frostschutz==1) && (Heizanforderung>0) && (Luefter_An))

 {
     Anfahrschaltung_Erfolgreich=0;
     SekundenCounter=0;
     Anfahrschaltung_Start=0;
     Anfahrschaltung_Aktiv=0;
     Starte_Anfahrschaltung=1;
     Versuche_Anfahrschaltung=0;
     Luefter_An=0;
     FL_SEND_POWER = 1;
     FLS_FREI1 = 1;
     return;
 }

//*****************************************************************************************************************
```

```
                                                    // Heizanforderung bei Außentemperatur kleiner 7°C ->
                                                    Anfahrschaltung starten
if (Heizanforderung>0 && Frostschutz==1 && Versuche_Anfahrschaltung==0 && !Luefter_An && Starte_Anfahrschaltung==0
&& Anfahrschaltung_Erfolgreich==0)
 {
    INT_FSF1_0 = 0;                                 // Debug-Information
    Starte_Anfahrschaltung=1;
    Anfahrschaltung_Start=0;
    Anfahrschaltung_Aktiv=0;
    Anfahrschaltung_Erfolgreich=0;
    SekundenCounter=0;
    Versuche_Anfahrschaltung=1;
    return;
 }

//*************************************************************************************************************
                                                    // Heizen  mit Frostschutz
 if (Anfahrschaltung_Erfolgreich==1 && Frostschutz==1)
 {
  INT_FSF1_0 = 7;                                   // Debug-Information
  if (Heizanforderung>0) //Heizen ist priorisiert
  {
  if (!Pumpe_An)                                    // Wenn die Pumpe noch aus ist, dann einschalten
  {
   Pumpe_An=1;
   FL_SEND_PUMPE = 1;
  }
  if (!Luefter_An)                                  // Wenn die Lüfter noch aus sind, dann einschalten
  {
   Luefter_An=1;
   FL_SEND_POWER = 1;
  }
  temp= L_LIB_READEE(EE_HeizVentilModus);
  if (HeizVentilModus!=temp)
  {
   HeizVentilModus = temp;
   FL_SEND_MODHEIZ = 1;
  }
  if ((HeizVentilModus==0) && (HeizVentilStellung!=255))
  {
   HeizVentilStellung = 255;                        // voll aufmachen, da Ventil nicht vernünftig proportional
                                                    regelt
   FL_SEND_VENHEIZ = 1;
  }

  if ((KuehlVentilStellung>0) || (KuehlVentilModus!=0))
  {
   KuehlVentilModus=0;
   KuehlVentilStellung = 0;
   FL_SEND_MODKUHL = 1;
   FL_SEND_VENKUHL = 1;
  }
  setVentilatoren (Heizanforderung);
  }
 }
                                                    // Heizen ohne Frostschutz
 if (Heizanforderung>0 && Frostschutz==0)           // Heizen ist priorisiert
 {
  if (!Pumpe_An)  // Wenn die Pumpe noch aus ist, dann einschalten
  {
   Pumpe_An=1;
   FL_SEND_PUMPE = 1;
  }
  if (!Luefter_An)                                  // Wenn die Lüfter noch aus sind, dann einschalten
  {
   Luefter_An=1;
   FL_SEND_POWER = 1;
  }
  temp= L_LIB_READEE(EE_HeizVentilModus);
  if (HeizVentilModus!=temp)
  {
   HeizVentilModus = temp;
   FL_SEND_MODHEIZ = 1;
  }
  if ((HeizVentilModus==0) && (HeizVentilStellung!=255))
  {
```

```
    HeizVentilStellung = 255;                    // voll aufmachen, da Ventil nicht vernünftig proportional regelt
    FL_SEND_VENHEIZ = 1;
   }
   if ((KuehlVentilStellung>0) || (KuehlVentilModus!=0))
   {
    KuehlVentilModus=0;
    KuehlVentilStellung = 0;
    FL_SEND_MODKUHL = 1;
    FL_SEND_VENKUHL = 1;
   }
   setVentilatoren (Heizanforderung);
  }
                                                 // Kühlen ohne Frostschutz
  if (Kuehlanforderung>0 && Frostschutz==0)
  {
    if (!Pumpe_An)                               // Wenn die Pumpe noch aus ist, dann einschalten
    {
     Pumpe_An=1;
     FL_SEND_PUMPE = 1;
    }
    if (!Luefter_An)                             // Wenn die Lüfter noch aus sind, dann einschalten
    {
     Luefter_An=1;
     FL_SEND_POWER = 1;
    }
    temp = L_LIB_READEE(EE_KuehlVentilModus);
    if (KuehlVentilModus!=temp)
    {
     KuehlVentilModus = temp;
     FL_SEND_MODKUHL = 1;
    }
    if ((KuehlVentilModus==0) && (KuehlVentilStellung!=255))
    {
     KuehlVentilStellung = 255;                  // voll aufmachen, da Ventil nicht vernünftig proportional
                                                 regelt
     FL_SEND_VENKUHL = 1;
    }
    if ((HeizVentilStellung>0) || (HeizVentilModus!=0))
    {
     HeizVentilModus=0;
     HeizVentilStellung = 0;
     FL_SEND_MODHEIZ = 1;
     FL_SEND_VENHEIZ = 1;
    }

    setVentilatoren (Kuehlanforderung);
   }
                                                 // Kühlen mit  Frostschutz nicht erlaubt
   if (Kuehlanforderung>0 && Frostschutz==1)
   {
    if (Pumpe_An)                                // Wenn die Pumpe noch an ist, dann abschalten
    {
     Pumpe_An=0;
     FL_SEND_PUMPE = 1;
    }
    if (Luefter_An)                              // Wenn die Lüfter noch an sind, dann abschalten
    {
     Luefter_An=0;
     FL_SEND_POWER = 1;
    }
    if ((Luefter_Abluft>0) || (Luefter_Zuluft>0))  // Wenn die Lüfterdrehzahl noch auf >0 eingestellt ist, dann auf
                                                 0 stellen
    {
     Luefter_Abluft = 0;
     Luefter_Zuluft = 0;
     FL_SEND_ABLUFT = 1;
     FL_SEND_ZULUFT = 1;
    }
    if ((HeizVentilStellung>0) || (HeizVentilModus!=0))  // Wenn das Heizventil noch offen ist, oder der
                                                 Heizventilmodus nicht auf Durchgriff steht, dann Modus auf
                                                 Durchgriff und Heizventil schließen
    {
     HeizVentilModus=0;
     HeizVentilStellung = 0;
     FL_SEND_MODHEIZ = 1;
```

```
    FL_SEND_VENHEIZ = 1;
   }
   if ((KuehlVentilStellung>0) || (KuehlVentilModus!=0))        // Wenn das Kuehlventil noch offen ist, oder der
                                                                Kuehlventilmodus nicht auf Durchgriff steht, dann Modus auf
                                                                Durchgriff und Kuehlventil schließen
   {
    KuehlVentilModus=0;
    KuehlVentilStellung = 0;
    FL_SEND_MODKUHL = 1;
    FL_SEND_VENKUHL = 1;
   }
   FLS_FREI1 = 1;
  }
                                                                // Lüften ohne Frostschutz

  if (Luftwechselrate_Anforderung>0 && Frostschutz==0)          // wenn Anforderung vom Lüftungsregler
  {
   if (!Luefter_An)                                             // Wenn die Lüfter noch aus sind, dann einschalten
   {
    Luefter_An=1;
    FL_SEND_POWER = 1;
   }

   setVentilatoren (Kuehlanforderung);
  }
                                                               // Weder Heizen noch Kühlen noch Lüften
  if (Kuehlanforderung==0 && Heizanforderung==0 && Luftwechselrate_Anforderung==0)
  {
   INT_FSF1_0 = 6;                                             // Debug-Information
   if (Pumpe_An)                                               // Wenn die Pumpe noch an ist, dann abschalten
   {
    Pumpe_An=0;
    FL_SEND_PUMPE = 1;
   }
   if (Luefter_An)                                             // Wenn die Lüfter noch an sind, dann abschalten
   {
    Luefter_An=0;
    FL_SEND_POWER = 1;
   }
   if ((Luefter_Abluft>0) || (Luefter_Zuluft>0))               // Wenn die Lüfterdrehzahl noch auf >0 eingestellt ist, dann auf
                                                               0 stellen
   {
    Luefter_Abluft = 0;
    Luefter_Zuluft = 0;
    FL_SEND_ABLUFT = 1;
    FL_SEND_ZULUFT = 1;
   }
   if ((HeizVentilStellung>0) || (HeizVentilModus!=0))         // Wenn das Heizventil noch offen ist, oder der
                                                               Heizventilmodus nicht auf Durchgriff steht, dann Modus auf
                                                               Durchgriff und Heizventil schließen
   {
    HeizVentilModus=0;
    HeizVentilStellung = 0;
    FL_SEND_MODHEIZ = 1;
    FL_SEND_VENHEIZ = 1;
   }
   if ((KuehlVentilStellung>0) || (KuehlVentilModus!=0))       // Wenn das Kuehlventil noch offen ist, oder der
                                                               Kuehlventilmodus nicht auf Durchgriff steht, dann Modus auf
                                                               Durchgriff und Kuehlventil schließen
   {
    KuehlVentilModus=0;
    KuehlVentilStellung = 0;
    FL_SEND_MODKUHL = 1;
    FL_SEND_VENKUHL = 1;
   }
   FLS_FREI1 = 1;
  }

}

//*********************************************************************
//**** 1s Timer    ***************************************************
//*********************************************************************
//
#pragma stackLevels 4
extern page1 void L_T01_FREI_0 (void)
```

```
#define     T01_FREI 1
{
 if (Versuche_Anfahrschaltung>3)
 {
  Frostschutz_Fehler=1;
  INT_FSF1_0 = 5;
  NewValue=1;
  return;
 }
 if (Starte_Anfahrschaltung==1 && Versuche_Anfahrschaltung<=3 && Anfahrschaltung_Erfolgreich==0 &&
Anfahrschaltung_Start==0)
 {
    SekundenCounter=0;
    Anfahrschaltung_Start=1;
    Starte_Anfahrschaltung=0;
    INT_FSF1_0 = 1;                                          // Debug-Information
    return;
 }
 if(Anfahrschaltung_Start==1 && Anfahrschaltung_Erfolgreich==0)
 {
     if (!Pumpe_An)                                          // Wenn die Pumpe noch aus ist, dann einschalten
     {
       Pumpe_An=1;
       FL_SEND_PUMPE = 1;
     }
     temp= L_LIB_READEE(EE_HeizVentilModus);
                                                              // Heizung aktivieren
     if (HeizVentilModus!=temp)
     {
       HeizVentilModus = temp;
       FL_SEND_MODHEIZ = 1;
     }
     if ((HeizVentilModus==0) && (HeizVentilStellung!=255))
     {
       HeizVentilStellung = 255;                              // voll aufmachen, da Ventil nicht vernünftig proportional regelt
       FL_SEND_VENHEIZ = 1;
     }
     //Kühlung deaktivieren
     if ((KuehlVentilStellung>0) || (KuehlVentilModus!=0))    // Wenn das Kuehlventil noch offen ist, oder der
                                                              Kuehlventilmodus nicht auf Durchgriff steht, dann Modus auf
                                                              Durchgriff und Kuehlventil schließen
     {
       KuehlVentilModus=0;
       KuehlVentilStellung = 0;
       FL_SEND_MODKUHL = 1;
       FL_SEND_VENKUHL = 1;
     }
     FLS_FREI1 = 1;
     Anfahrschaltung_Aktiv=1;
     Anfahrschaltung_Start=0;
     INT_FSF1_0 = 2;                                          // Debug-Information
     return;
 }
 if (Anfahrschaltung_Aktiv==1 && Anfahrschaltung_Erfolgreich==0)
 {
     if (WW_Ruecklauftemperatur>Min_RL_Temp_Anfahren)
     {
       Anfahrschaltung_Erfolgreich=1;
       Anfahrschaltung_Aktiv=0;
       Versuche_Anfahrschaltung=0;
       Anfahrschaltung_Start=0;
       Starte_Anfahrschaltung=0;
       Anfahrschaltung_Aktiv=0;
       SekundenCounter=0;
       NewValue=1;
       INT_FSF1_0 = 3;                                        // Debug-Information
     }
     else
     {
       SekundenCounter++;
       if (SekundenCounter<Anfahrzeit)
       {
         return;
       }
       Anfahrschaltung_Erfolgreich=0;
```

```
      SekundenCounter=0;
      Versuche_Anfahrschaltung++;
      Anfahrschaltung_Start=0;
      Anfahrschaltung_Aktiv=0;
      Starte_Anfahrschaltung=1;
      NewValue=1;
      INT_FSF1_0 = 4;                                   // Debug-Information
    }
  }

}
//
//*********************************************************************
//**** Telegrammeingang 1 ********************************************
//*********************************************************************
// Eingang   : BEF_WDH= Daten-Byte 1 (High)    (nicht veraendern!)
//             BEF_WDL= Daten-Byte 2 (Low)     (nicht veraendern!)
//             BEF_WAH= Adress-Byte 1 (High)   (nicht veraendern!)
//             BEF_WAL= Adress-Byte 2 (Low)    (nicht veraendern!)
//             HELP1  = Selektor aus Empfangsliste, inkl. Parameter i: "o0iiixxx"
//*********************************************************************
//
#pragma stackLevels 5
extern page1 void L_TELIN_FREI1_0 (void)
#define      TELIN_FREI1          1
{
  temp = HELP1;
  temp = temp>>3; //schiebt die Don't Care Bits raus
  temp = temp&7;  // nur untere 3bits sind interessant

  switch (temp)
  {
    case 0:  //  Heizanforderung
      if ((BEF_WDH!=2)&& (!HELP1.7))
        return; // Falscher Datentyp
      Heizanforderung =  BEF_WDL;
      NewValue=1;
      break;

    case 1:  // Kuehlanforderung
      if ((BEF_WDH!=2)&& (!HELP1.7))
        return; // Falscher Datentyp
      Kuehlanforderung =  BEF_WDL;
      NewValue=1;
      break;

    case 2:  //WW_Ruecklauftemperatur
      if ((!BEF_WDH.7) || (BEF_WDH.6))
        return; // Falscher Datentyp
      WW_Ruecklauftemperatur.low8 =  BEF_WDL;
      WW_Ruecklauftemperatur.high8 = BEF_WDH &~192;
      if (BEF_WDH.5)
      WW_Ruecklauftemperatur.high8 |= 192U;             // Korrektur 2er Komplement von 14 Bit auf 16 Bit
      NewValue=1;
      break;

     case 3:  //Außenluft
      if ((!BEF_WDH.7) || (BEF_WDH.6))
        return; // Falscher Datentyp
      Aussen_Temperatur.low8 =  BEF_WDL;
      Aussen_Temperatur.high8 = BEF_WDH &~192;
      if (BEF_WDH.5)
        Aussen_Temperatur.high8 |= 192U;                // Korrektur 2er Komplement von 14 Bit auf 16 Bit
      NewValue=1;
      break;

      case 4:  // Zustand Pumpe
      if (BEF_WDH!=0)
        return; // Falscher Datentyp
      Zustand_Pumpe =  BEF_WDL.0;
      NewValue=1;
      break;

      case 5:  // Luftwechselrate
      if ((BEF_WDH!=2)&& (!HELP1.7))
        return; // Falscher Datentyp
```

```
    Luftwechselrate_Anforderung =  BEF_WDL;
    NewValue=1;
    break;

  }
}
//
//*************************************************************************
///**** Telegrammversand 1 ***************************************************
//*************************************************************************
// Ausgabe   : HELP1= Byte 1 (High)
//           HELP2= Byte 2 (Low)
//           w    = Telegrammversandsteuerwort (C_SEND_SOND1 + ...)
//*************************************************************************
//
#pragma stackLevels 5
extern page1 uns8 L_SEND_FREI1_0 (void)
#define     SEND_FREI1          1
{
 if (FLB_SNALL_FSF_0)
 {
  FLB_SNALL_FSF_0 = 0;                              //Aufruf loeschen
  SEND_FLAGS = MASK_SEND_FSFALL;
 }
 SEND_FLAGS |=SEND_TXT;                             //abgebrochenen Versand ebenso erneut anfordern
 SEND_TXT = 0               ;
  //*******************************
 if (FL_SEND_PUMPE)                                 // Pumpe_An
 {
  FL_SEND_PUMPE=0;
  FL_SENDTXT_PUMPE = 1;
  HELP1  = 0;                                       // Typ = Binaer
  if (Pumpe_An)
    HELP2 = 1;
  else
    HELP2 = 0;
  return C_SEND_FREI1         ;
 }
 //*******************************
  if (FL_SEND_POWER)                                // Luefter_An
  {
    FL_SEND_POWER=0;
    FL_SENDTXT_POWER = 1;
    HELP1              = 0;                         // Typ = Binaer
    if (!Luefter_An) //Invertiert?
      HELP2 = 1;
    else
      HELP2 = 0;
    return C_SEND_FREI1+1    ;
  }
 //*******************************
    if (FL_SEND_ABLUFT)                             // Abluft
    {
      FL_SEND_ABLUFT=0;
      FL_SENDTXT_ABLUFT = 1;
      HELP1              = 2;                       // Typ = Verhaeltnis
      HELP2 = Luefter_Abluft; //
      return C_SEND_FREI1+2  ;
    }
       //*******************************
    if (FL_SEND_ZULUFT)                             // Zuluft
    {
      FL_SEND_ZULUFT=0;
      FL_SENDTXT_ZULUFT = 1;
      HELP1              = 2;                       // Typ = Verhaeltnis
      HELP2 = Luefter_Zuluft; //
      return C_SEND_FREI1+3  ;
    }
 //*******************************
     if (FL_SEND_MODHEIZ)                           // HeizVentilModus
     {
       FL_SEND_MODHEIZ=0;
       FL_SENDTXT_MODHEIZ = 1;
       HELP1           = 8;                         // Typ = Auswahl
       HELP2 = HeizVentilModus;
```

```
      return C_SEND_FREI1+4              ;
    }
 //*****************************
      if (FL_SEND_VENHEIZ)                                  // HeizVentilStellung
      {
       FL_SEND_VENHEIZ=0;
       FL_SENDTXT_VENHEIZ = 1;
       HELP1          = 2;                                  // Typ = Verhältnis
       HELP2 = HeizVentilStellung;
       return C_SEND_FREI1+5             ;
      }
 //*****************************
       if (FL_SEND_MODKUHL)                                 // KuehlVentilModus
       {
        FL_SEND_MODKUHL=0;
        FL_SENDTXT_MODKUHL = 1;
        HELP1         = 8;                                  // Typ = Auswahl
        HELP2 = KuehlVentilModus;
        return C_SEND_FREI1+6            ;
       }
 //*****************************
        if (FL_SEND_VENKUHL)                                // KuehlVentilStellung
        {
         FL_SEND_VENKUHL=0;
         FL_SENDTXT_VENKUHL = 1;
         HELP1       = 2;                                   // Typ = Verhältnis
         HELP2 = KuehlVentilStellung;
         return C_SEND_FREI1+7           ;
        }
 //*****************************
 SEND_FLAGS=0;                                               // Fehler!
 return C_SEND_NOT;
}
//
//*************************************************************************
///**** Telegrammversand 1 erfolgreich abgeschlossen ****************
//*************************************************************************
// Eingabe   : DAT_TX0_1= Adresse-H des zuletzt versandten Bustelegramms
//           DAT_TX1_1= Adresse-L des zuletzt versandten Bustelegramms
//           DAT_TX2_1= Byte-H des zuletzt versandten Bustelegramms
//           DAT_TX3_1= Byte-L des zuletzt versandten Bustelegramms
//*************************************************************************
//
#pragma stackLevels 5
extern page1 void L_SEND_FREI1_ENDE_0 (void)
#define     SEND_FREI1_ENDE               1
{
 SEND_TXT = 0       ;
 if (SEND_FLAGS)                                            // weitere Versandwuensche?
   FLS_FREI1 = 1;     ;                                     // ja -> weiter senden
}
//
//*************************************************************************
//**** Initialisierung ************************************************
//*************************************************************************
// Achtung! Diese Routine muss hinter den anderen Interface-Einsprungmarken
//   stehen, um die Defines "C_Sprungname" auszuwerten!
//
#pragma stackLevels 4
extern page1 void L_INIT_FREI_0 (void)
{
#define     FSF          1;                                 // Flag fuer Include: Dies ist eine freie Sonderfunktion
#include    <INITc.inc>;                                    // Setzen von Bezeichung, Versionsnummer und Einspruengen
//
UntererSkalierungsWert = L_LIB_READEE(EE_UntererSkalierungsWert);
Skalierungsfaktor = L_LIB_READEE(EE_Skalierungsfaktor);
SkalierungsfaktorAbluft = L_LIB_READEE(EE_SkalierungsfaktorAbluft);
Anfahrzeit = L_LIB_READEE(EE_Anfahrzeit)

}
```

A.3 Platinenlayout

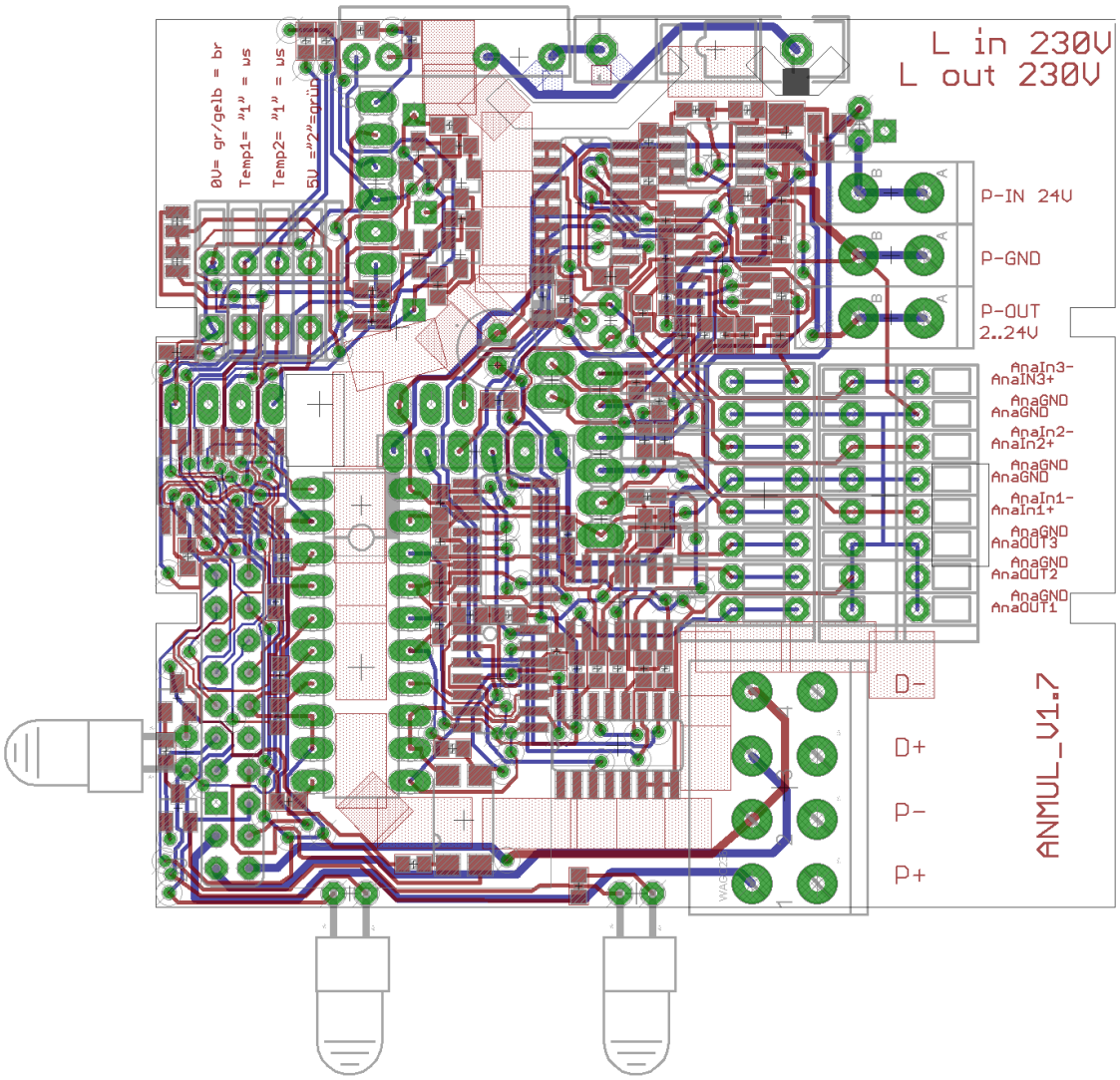


Abbildung A.2: Geometrische Anordnung der Bauteile des Anwendungsmoduls

Anhang B

Strukturbeschreibende Matrizen des realisierten Systems zur Klimaregulierung

B.1 Programm- und Gerätematrix

$$\underline{\underline{P}}=\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}$$

$$\underline{\underline{G}}=\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}$$

B.2 Inzidenzmatrizen eines kausalen Teilnetzes aus der Struktur des verteilten Steuerungsprozesses

$$\underline{\underline{C}}^{+}_{F,S,K,red} = \begin{bmatrix} \begin{pmatrix} 1&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&1 \end{pmatrix} & \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{\mathbf{0}}} & \begin{pmatrix} 1&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&1 \end{pmatrix} & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} & \begin{pmatrix} 1&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&1 \end{pmatrix} \end{bmatrix}$$

$$\underline{\underline{C}}^{-}_{F,S,K,red} = \begin{bmatrix} \begin{pmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0 \end{pmatrix} & \begin{pmatrix} 1&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0 \end{pmatrix} & \begin{pmatrix} 0&0&0&0&0&0&0&0\\ 1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0 \end{pmatrix} \\ \begin{pmatrix} 0&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&1&0&0&0&0&0 \end{pmatrix} & \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{\mathbf{0}}} & \begin{pmatrix} 0&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&1&0&0&0&0&0 \end{pmatrix} & \underline{\underline{\mathbf{0}}} \end{bmatrix}$$

B.3 Inzidenzmatrizen eines Teilnetzes zur Bewertung der Sequenzdetektierbarkeit

$$\underline{\underline{C}}^{+}_{TF,K,TP,red} = \begin{bmatrix} \begin{pmatrix} 1&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&1 \end{pmatrix} & \underline{\underline{\mathbf{0}}} & \begin{pmatrix} 0&0&0&0&0\\ 1&0&0&0&0\\ 0&1&0&0&0\\ 0&0&1&0&0\\ 0&0&0&1&0\\ 0&0&0&0&0\\ 0&0&0&0&0\\ 0&0&0&0&1\\ 0&0&0&0&1\\ 0&0&0&0&1\\ 0&0&0&0&1\\ 0&0&0&0&1\\ 0&0&0&0&0\\ 0&0&0&0&0 \end{pmatrix} & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{\mathbf{0}}} & \begin{pmatrix} 1&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&1 \end{pmatrix} & \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} & \begin{pmatrix} 1&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&1&1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&1&1&1&1&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&1&1&1&1&1&1 \end{pmatrix} \end{bmatrix}$$

$$\underline{\underline{C}}^{-}_{TF,K,TP,red} = \begin{bmatrix} \begin{pmatrix} 0&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0 \end{pmatrix} & \begin{pmatrix} 0&0&0&0&0&0&0&0\\ 1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0 \end{pmatrix} & \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} \\ \begin{pmatrix} 0&0&1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0&0&0&0&1&0&0 \end{pmatrix} & \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{\mathbf{0}}} & \underline{\underline{\mathbf{0}}} & \begin{pmatrix} 1&0&0&0&0\\ 0&1&0&0&0\\ 0&0&1&0&0\\ 0&0&0&1&0\\ 0&0&0&0&1 \end{pmatrix} & \underline{\underline{\mathbf{0}}} \end{bmatrix}$$